University of Granada
Ph.D. Program in Mathematics and Statistics

STATISTICAL ANALYSIS OF SPATIO-TEMPORAL CRIME PATTERNS: OPTIMIZATION OF PATROLLING STRATEGIES

Miguel Camacho Collados

Ph.D. Thesis supervised by:

José Miguel Angulo Ibáñez
Federico Liberatore

Granada, July 2016

Editor: Universidad de Granada. Tesis Doctorales
Autor: Miguel Camacho Collados
ISBN: 978-84-9163- 008-1
URI: http://hdl.handle.net/10481/44557

El doctorando Miguel Camacho Collados y los directores de la tesis José Miguel Angulo Ibáñez y Federico Liberatore, garantizamos, al firmar esta tesis doctoral, que el trabajo ha sido realizado por el doctorando bajo la dirección de los directores de la tesis y, hasta donde nuestro conocimiento alcanza, en la realización del trabajo, se han respetado los derechos de otros autores a ser citados cuando se han utilizado sus resultados o publicaciones.

Granada, julio de 2016

Directores de la Tesis:

Fdo: José Miguel Angulo Ibáñez Fdo: Federico Liberatore

Doctorando
Fdo: Miguel Camacho Collados

ACKNOWLEDGMENTS

First of all, I would like to express my sincere gratitude to my supervisors Dr. José Miguel Angulo and Dr. Federico Liberatore for the continuous support to my Ph.D. studies and related research, for their expertise, understanding, patience and immense motivation. Their guidance helped me in all the time of research and writing of this thesis. I could not have expected having better supervisors and mentors for my Ph.D. studies.

Besides my supervisors, I would like to thank Prof. Andrea Bertozzi, for her insightful comments and encouragement, as well for her hospitality and all the opportunities she provided, which incented me to widen my research from various perspectives. A big thank also to the UCLA research team, especially to Prof. Jeff Brantingham and to Commander Sean Malinowski (Los Angeles Police Department).

My sincere thanks also goes to the Fulbright Comission, who provided me with an incredible opportunity.

I would like as well to express my gratitude to the Spanish National Police, for facilitating all this process.

Also, I am extremely grateful to my family for all the support and love they constantly gave me.

In conclusion, I recognize that this research would not have been possible without the financial assistance of the University of Granada,

the Spanish National Police, the Fulbright Grant, the National Police Foundation Scholarship, the UCLA (University of California Los Angeles). All the financial help is gratefully acknowledged.

ABBREVIATIONS

ANOVA - Analysis Of Variance

CRFU - Crime Risk Forecasting Unit

DPPU - Data Pre-Processing Unit

DSS - Decision Support System

GIS - Geographic Information Systems

GRASP - Greedy Randomized Adaptive Search Procedure

LEMAS - Law of Enforcement Management and Administrative Statistics

MC - PDP - Multi-Criteria Police Districting Problem

MSE - Mean Squared Error

P3-DSS - Predictive Police Patrolling Decision Support System

PDP - Police Districting Problem

PSOU - Patrol Sector Optimization Unit

RTM - Risk Terrain Modeling

SDHC - Steepest Descent Hill Climbing

SHC - Simple Hill Climbing

SNPC - Spanish National Police Corps

TS - Tabu Search

SYNOPSIS

During most of the 20th century, police districts were drawn by police officers on road maps with markers, just by following the major streets in the area, without making too much of an effort to accomplish geographic or workload balance. Since 1972, a number of mathematical optimization models have been proposed to serve this purpose and the Police Districting Problem (PDP) was born. The PDP aims at partitioning the territory under the jurisdiction of a Police Department in the best possible way, with respect to several time, cost, performance, and topological attributes. Only after the recent advancement of Geographic Information Systems (GIS) and computer technology, which allowed for reasonable computational time and ease of representation and manipulation, automatic methodologies for the definition of police districts gained popularity among practitioners. However, studies integrating GIS and sophisticated mathematical modeling for police districting remain a rarity, and the "map-and-marker" method is still one of the most applied redistricting procedures. Nevertheless, the importance of a balanced definition of the police districts is unquestioned and the implementation of decision-aid tools for the allocation of police resources has proven to be extremely beneficial. In fact, all the works report a dramatic improvement in workload distribution compared to handmade districts which, in turn, results in enhanced performances

and efficiency.

In Spain, the security of towns is borne by the Spanish National Police Corps (SNPC), usually sharing territory with other local security forces. The SNPC is an armed Institute of civil nature, subordinated to the Ministry of Home Affairs. Among its duties are: keeping and restoring order and public safety and to prevent the commission of criminal acts. The SNPC is one of the country's most valued institutions and is located at the global forefront in the fight against crime, with the aim of constant innovation.

To improve the effectiveness of patrolling operations and increase the efficiency in the use of resources, the SNPC has started to develop a Decision Support System (DSS) comprising tools and models to assist various public security tasks. One of the main objectives of the system is the implementation of a predictive patrolling policy to increase the presence of agents in the areas where they are most needed, to reduce the probability of occurrence of crime. To this end, the author, in collaboration with professionals from the SNPC, developed a Predictive Policing tool for crime risk forecasting based on the statistical analysis of spatio-tempooral crime patterns, and an optimization model for the definition of patrolling sectors configuration, tailored to suit the requirements of the SNPC.

The first contribution of the investigation is a Multi-Criteria Police Districting Problem (MC-PDP) for the efficient and effective design of patrol sectors. The goal is to partition the territory under the jurisdiction of a district into patrol sectors in the best possible way. The criteria for evaluating the goodness of the configurations of the patrol sectors were identified after interviewing several service coordinators

and a number of agents involved in public safety operations. The result is a mathematical optimization model which finds an efficient configuration in terms of prevention service and attention to calls, distributing the workload equitably among agents. The model proposed is multi-criteria in nature. Given the non-linear nature of its restrictions, the author proposes for its solution a local search heuristic algorithm. A case study on the Central District of Madrid is presented and the performance of the algorithm is assessed. The author shows empirically that the algorithm generates rapidly patrolling configurations that are more efficient than those currently employed by the SNPC.

The second contribution is a Decision Support System (DSS) that can help to optimize the efficient use of the scarce human resources available is investigated. A DSS that merges Predictive Policing capabilities with a Patrolling Districting Model is presented, for the design of predictive patrolling areas. The proposed DSS, developed in close collaboration with the SNPC, defines partitions of the territory under the jurisdiction of a district that are efficient and balanced at the same time, according to the preferences of a decision maker. To analyze the crime records provided by the SNPC, a methodology for the description of spatially and temporally indeterminate crime events has been developed. The results of the experiments show that the proposed DSS clearly outperforms the patrolling area definitions currently in use by the SNPC. To compare the solutions in terms of efficiency loss, the author discusses how to build an operational envelope for the problem considered, which can be used to identify the range of performances associated with different patrolling strategies.

The third contribution is the extention of the MC-PDP to gen-

erate efficient convex partitions on generic graphs, which increases the practical usefulness and applicability of the model. Also, the author proposes and compares three local search algorithms and tests them on real crime data from the Central District of Madrid. Local search algorithms move from solution to solution in the space of candidate solutions (the search space) by applying local changes, until certain termination criteria are satisfied, e.g., a solution deemed optimal is found or a time bound is elapsed. One of the main advantages of local search algorithms is that they are anytime algorithms, which means that they can return a valid solution even if they are interrupted at any time before they end. For this reason, they are often used to tackle hard optimization problems in a real-time environment, such as the MC-PDP. Different implementations of termination criteria and area of search in different local search algorithms are described. For Simple Hill Climbing (SHC) at each iteration, the algorithm explores the neighborhood of the incumbent solution to find a better one. The neighborhood of a solution is the set of solutions that can be obtained from the current one by changing it slightly. The algorithm terminates when no improving solution is found or the time limit is exceeded. The Steepest Descent Hill Climbing (SDHC) algorithm is a variant of the SHC that explores the whole neighborhood of the incumbent solution and chooses the best solution belonging to it. A Tabu Search algorithm (TS), similarly to the SDHC, explores the whole neighborhood of the incumbent solution. However, the TS chooses for the next iteration the best solution found that is not tabu. Also, the TS does not terminate if an improving solution is not found. This allows the algorithm to escape local optima. The criterion that is used to declare a certain

point of the neighborhood as tabu is based on a short-term memory. During the exploration of a neighborhood all the solutions found that are already included in the short term memory are marked as tabu and their expiration counter are reset to initial set. Finally, at the end of the iteration, all the expiration counters are decreased by one and the solutions whose counters have reached zero are removed from the short term memory. The algorithm terminates when the time limit is exceeded, when no non-tabu solution is found in the current neighborhood, or after a fixed number of non-improving iterations. The results of the computational analysis show that the TS presented in this part produces solutions that are on average better than those identified by the SDHC or SHC algorithms. Here, the author offers new interesting lines to be pursued. In terms of modeling, solving the MC-PDP on a graph simplifies the inclusion of demographic data in the model, such as the racial composition of a census district. This makes the DSS more practical and realistic and further suggestions for research are provided.

SINOPSIS

Durante la mayor parte del Siglo XX, los agentes de policía dibujaban con rotuladores los distritos de policía en los mapas de carretera, simplemente siguiendo las principales calles de la zona, sin hacer demasiado esfuerzo en lograr un equilibrio en la distribución de áreas de patrulla o la carga de trabajo. Desde 1972, se propusieron una serie de modelos matemáticos de optimización para servir a este propósito y así fue como nació el Problema de Distribución de Distritos Policiales (PDDP). El PDDP tiene como objetivo dividir el territorio bajo la jurisdicción de un departamento de policía de la mejor manera posible, con respecto a variaciones en tiempo, coste, rendimiento y características topológicas. Sólo después del reciente avance en los Sistemas de Información Geográfica (SIG) y la tecnología informática, que permitió tener un tiempo razonable de cálculo, facilidad de representación y manipulación; es cuando las metodologías automáticas para la definición de los distritos policiales comienzan a ganar popularidad entre los profesionales. Sin embargo, los estudios que integran SIG y los modelos matemáticos sofisticados para la división en distritos de la policía siguen siendo extraños, y el método de mapa-y-rotulador sigue siendo uno de los procedimientos de redistribución de distritos más aplicados. En cualquier caso, la importancia de una definición equilibrada de los distritos de policía es incuestionable y la implementación

de herramientas de ayuda a la decisión para la asignación de recursos de la policía ha demostrado ser muy beneficiosa. De hecho, todos los informes señalan una increíble mejora en la distribución de la carga de trabajo en comparación con los distritos hechos a mano, que, a su vez, se traduce en un mayor rendimiento y eficiencia.

En España, la seguridad de los núcleos urbanos es responsabilidad del Cuerpo Nacional de Policía (CNP), generalmente compartiendo territorio con otras fuerzas de seguridad locales. El CNP es un instituto armado de naturaleza civil dependiente del Ministerio de Interior. Entre sus funciones están: mantener y restaurar el orden y la seguridad pública y prevenir la comisión de actos delictivos. El CNP es una de las instituciones más valoradas del país y se encuentra a la vanguardia internacional en la lucha contra la delincuencia, con un objetivo de innovación constante.

Para mejorar la eficacia de las operaciones de patrullaje y aumentar la eficiencia en el uso de los recursos, El CNP ha comenzado a desarrollar un Sistema de Soporte a la Decisión (SSD) que comprende herramientas y modelos para ayudar a diversas tareas de seguridad pública. Uno de los principales objetivos del sistema es la implementación de una política de patrullaje predictivo para aumentar la presencia de los agentes en las zonas donde más se necesitan, y así reducir la probabilidad de ocurrencia del delito. Para tal fin, el autor, en colaboración con profesionales del CNP, desarrolló una herramienta Policial Predictiva para el pronóstico de riesgo de delitos basado en el análisis estadístico de los patrones de criminalidad espacio-temporal, y un modelo de optimización para la definición de la configuración de sectores de patrullaje, adaptado a los requisitos del CNP.

La primera contribución de la investigación es un (MC-PDDP) Problema Multi-Criterio de División de Distritos Policiales para el diseño eficiente y eficaz de los sectores de patrulla. El objetivo es dividir el territorio bajo la jurisdicción de un distrito en sectores de patrulla de la mejor manera posible. Se identificaron los criterios para evaluar la bondad de las configuraciones de los sectores de patrulla después de entrevistar a varios coordinadores de servicio y un número de agentes implicados en las operaciones de seguridad pública. El resultado es un modelo de optimización matemática con el que se logra una configuración eficiente en términos de servicio y atención a la prevención de las llamadas, así como la distribución de la carga de trabajo de manera equitativa entre los agentes. El modelo propuesto es de naturaleza multi-criterio. Dada la naturaleza no lineal de sus restricciones, el autor propone para su solución un algoritmo heurístico de búsqueda local. Se presenta un caso de estudio en el Distrito Central de Madrid y el rendimiento del algoritmo es evaluado. El autor demuestra empíricamente que el algoritmo genera rápidamente configuraciones de patrulla, que son más eficientes que las actualmente empleados por el CNP. La segunda contribución es un Sistema de Soporte a la Decisión (SSD) que puede ayudar a optimizar el uso eficiente de los escasos recursos humanos disponibles. Se muestra un SSD que combina las Capacidades Predictivas Policiales con un Modelo de Patrullaje en Distritos, para el diseño de las zonas de patrullaje predictivo. El SSD propuesto, desarrollado en estrecha colaboración con el CNP, define las divisiones del territorio bajo la jurisdicción de un distrito que sea eficiente y equilibrado al mismo tiempo, de acuerdo a las preferencias de quien tome las decisiones. Para el análisis de los registros de deli-

tos previstos por el CNP, se ha desarrollado una metodología para la descripción de los eventos delictivos espacial y temporalmente indeterminados. Los resultados de los experimentos muestran que el SSD propuesto mejora claramente las definiciones de área de patrullaje que actualmente son usadas por el CNP. Para comparar las soluciones en términos de pérdida de eficiencia el autor discute cómo construir una solución operativa para el problema considerado que pueda utilizarse para identificar la gama de actuaciones relacionadas con diferentes estrategias de patrullaje.

La tercera contribución es la extensión del MC-PDDP para generar divisiones convexas eficientes en los gráficos genéricos, lo que aumenta la utilidad práctica y la aplicabilidad del modelo. Además, el autor propone y compara tres algoritmos de búsqueda local, y los pone a prueba con los datos reales de la delincuencia en el Distrito Central de Madrid. Los algoritmos de búsqueda local se mueven de solución a solución en el espacio de soluciones de candidatos (el espacio de búsqueda) mediante la aplicación de cambios locales, hasta que ciertos criterios de terminación estén satisfechos; por ejemplo, una solución considerada óptima es encontrada o transcurre un período determinado de tiempo. Una de las principales ventajas de los algoritmos de búsqueda local es que son algoritmos en cualquier momento, lo que significa que pueden devolver una solución válida incluso si se interrumpe en cualquier momento antes de que terminen. Por esta razón, a menudo se utilizan para hacer frente a problemas de optimización difíciles en un entorno en tiempo real, como el MC-PDDP. Se describen diferentes implementaciones de criterios de terminación y el área de búsqueda en diferentes algoritmos de búsqueda local. Para Simple Hill

Climbing (SHC), en cada iteración, el algoritmo explora los alrededores de la posible solución para encontrar una mejor. El vecindario de una solución es el conjunto de soluciones que se pueden obtener a partir de la actual, cambiándola ligeramente. El algoritmo termina cuando no se encuentra solución de mejora o se excede el límite de tiempo. El algoritmo del Steepest Descent Hill Climbing (SDHC) es una variante del SHC que explora todo el vecindario de la posible solución y elige la mejor solución perteneciente a la misma. Un algoritmo de búsqueda tabú (BT), de manera similar al DSSC, explora todo el universo de la solución posible. Sin embargo, el BT elige para la siguiente iteración que la mejor solución encontrada que no sea tabú. Además, el BT no termina si no se encuentra una solución de mejora. Esto permite al algoritmo salir del óptimo local. El criterio que se utiliza para nombrar un cierto punto del vecindario como tabú se basa en una memoria a corto plazo. Durante la exploración de un vecindario todas las soluciones encontradas que ya están incluidas en la memoria a corto plazo se marcan como tabú y su contador de caducidad se restablece a la configuración inicial. Por último, al final de la iteración, todos los contadores de caducidad se reducen uno solo y las soluciones cuyos contadores han llegado a cero se eliminan de la memoria a corto plazo. El algoritmo termina cuando se supera el límite de tiempo, cuando no hay solución no tabú en el vecindario actual, o cuando después de un número fijo de iteraciones no hay mejora. Los resultados del análisis computacional muestran que el BT presentado en esta parte produce soluciones que son, en promedio, mejor que las identificadas por el SDHC o algoritmos de SHC. Aquí, el autor ofrece nuevas líneas de interés para ser investigados. En cuanto a la modelización, la resolución

de MC-PDDP en un gráfico simplifica la inclusión de datos demográficos en el modelo, como la composición racial de un distrito censal. Esto hace el SSD más práctico y realista, y se proporcionan sugerencias adicionales para la investigación.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

For most of the 20th century, police districts have been drawn by police officers on a road map with a marker, by simply following the major streets in the area, without making a significant effort to accomplish geographic or workload balance [14]. Since the seminal paper by Mitchell [71], a number of mathematical optimization models have been proposed and the Police Districting Problem (PDP) was born. The PDP aims at partitioning the territory under the jurisdiction of a Police Department in the best possible way, with respect to several time, cost, performance, and topological attributes. Use of automatic methodologies for the definition of police districts only became popular among the practitioners, after the recent advances in Geographic Information Systems (GIS) and computer technology, that enabled reasonable computational times and ease of representation and manipulation [105]. However, studies integrating GIS and sophisticated mathematical modeling for police districting remain a rarity [14], and the "map-and-marker method" is still one of the most widely used redistricting procedures. Nevertheless, the importance of a balanced definition of the police districts is unquestioned and the implementation of tools for aiding in making the decisions for the allocation of police resources has proven to be extremely beneficial, as shown by the substantial academic literature on this topic in the last decades [31]. In fact, all the works report a dramatic improvement in workload distribution compared to hand-made districts which, in turn, results in enhanced performance and efficiency.

In Spain, the security of towns is the responsibility of the Spanish National Police Corps (SNPC), usually sharing territory with other local security forces. The SNPC is an armed institute of a civil nature, dependent on the Ministry of Home Affairs. Among its duties are: keeping and restoring order and public safety

and preventing the commission of criminal acts. The SNPC is one of the country's most valued institutions and is located at the global forefront of the fight against crime, with the aim of constant innovation. In recent years, the socio-economic context in Spain has been that of a serious crisis, which has reduced the resources and the number of police officers available to the SNPC. In order to continue providing the same level of security, the SNPC is taking cutting-edge steps to increase its competitiveness. Under the current system, the distribution of patrols is the responsibility of the inspectors who, under normal conditions, locate the agents according to the neighborhood borders defined more than 50 years ago. To improve the effectiveness of patrolling operations and increase the efficiency in the use of scarce resources, the SNPC has started to develop a Decision Support System (DSS) comprising tools and models to assist in various public security tasks [17]. One of the main objectives of the system is the implementation of a predictive patrolling policy to increase the presence of agents in the areas where they are most needed, to reduce the probability of the occurrence of crime. To this end, the author helped in developing, in collaboration with professionals from the SNPC, an optimization model for the definition of patrolling sector configurations, tailored to suit the requirements of the SNPC. Since the model is required to be included in the DSS, it is expected to be interactive. Thus, we implemented a heuristic algorithm that provides acceptable solutions quickly. By combining the proposed algorithm with a crime risk forecasting model [80, 97], a predictive patrolling system is obtained. For the SNPC, the implementation of a predictive patrolling system also represents a paradigm shift, from detention to prevention, resulting in reductions in the costs of detention and an improvement in the actual, subjective, and social level of safety.

The remainder of this thesis is organizes as follows. Chapter 2 reviews the literature on the PDP, presents the paradigm of Predictive Policing, and analizes the development and use of new technological advances, including DSS, in the SNPC. The problem of partitioning a territory represented as a matrix into homogeneous convex sectors is presented in Chapter 3. In Chapter 4 the optimization model is embedded into a comprehensive DSS for the implementation of a Smart Patrolling paradigm. Chapter 5 extends the original model to be solved on a generic graph and proposes and compares more efficient solution algorithms. The proposed algorithms are tested on a real case study in Chapter 6. Finally, the thesis is concluded with Chapter 7 that presents a summary of the main contribution

of this work, as well as some guidelines for future research.

CHAPTER 1

INTRODUCCIÓN

Durante la mayor parte del Siglo XX, los agentes policiales dibujaban con rotulador los distritos de policía en los mapas de carretera, siguiendo las principales calles de la zona, sin poner demasiado esfuerzo en lograr un equilibrio geográfico o de carga de trabajo [14]. Desde el trabajo de Mitchell [71], se propuso una serie de modelos matemáticos de optimización para servir a este propósito y así fue como nació el Problema de Distribución de Distritos Policiales (PDDP). El PDDP tiene como objetivo dividir el territorio bajo la jurisdicción de un departamento de policía de la mejor manera posible, con respecto a variaciones en tiempo, coste, rendimiento y características topológicas. Sólo después del reciente avance en los Sistemas de Información Geográfica (SIG) y la tecnología informática, que permitió un tiempo razonable de cálculo, facilidad de representación y manipulación; es cuando las metodologías automáticas para la definición de los distritos policiales comienzan a ganar popularidad entre los profesionales [105]. Sin embargo, los estudios que integran SIG y modelos matemáticos sofisticados para la división en distritos de la policía siguen siendo extraños [14], y el método de mapa-y-rotulador sigue siendo uno de los procedimientos de redistribución de distritos más aplicados. No obstante, la importancia de una definición equilibrada de los distritos de policía es incuestionable y la implementación de herramientas de ayuda a la decisión para la asignación de recursos de la policía ha demostrado ser muy beneficioso, como es reflejado en la abundante literatura académica sobre este tema en las últimas décadas [31]. De hecho, todos los informes señalan una increíble mejora en la distribución de la carga de trabajo en comparación con los distritos hechos a mano, que, a su vez, se traduce en un mayor rendimiento y eficiencia. En España, la seguridad de las ciudades es responsabilidad del Cuerpo Nacional de Policía (CNP), generalmente compartiendo territorio con otras fuerzas de seguridad locales. El

CNP es un instituto armado de naturaleza civil dependiente del Ministerio de Interior. Entre sus funciones están: mantener y restaurar el orden y la seguridad pública y la de prevenir la comisión de actos delictivos. El CNP es una de las instituciones más valoradas del país y se encuentra a la vanguardia mundial en la lucha contra la delincuencia, con el objetivo de innovación constante. En los últimos años, el contexto socio-económico en España ha sido el de una grave crisis, que ha reducido los recursos y el número de agentes de policía u oficiales disponibles para el CNP. Con el fin de continuar proporcionando el mismo nivel de seguridad el CNP está tomando medidas de vanguardia para aumentar su competitividad. Bajo el sistema actual, la distribución de las patrullas es responsabilidad de los inspectores que, en condiciones normales, ubican a los agentes de acuerdo con las fronteras de los barrios pautadas hace más de 50 años. Para mejorar la eficacia de las operaciones de patrullaje y aumentar la eficiencia en el uso de los recursos el CNP ha comenzado a desarrollar un Sistema de Soporte a la Decisión (SSD) que comprende herramientas y modelos para ayudar a diversas tareas de seguridad pública [17]. Uno de los principales objetivos del sistema es la implementación de una política de patrullaje predictivo para aumentar la presencia de los agentes en las zonas donde más se necesitan, y así reducir la probabilidad de ocurrencia del delito. Para tal fin, el autor, en colaboración con profesionales del CNP, desarrolló una herramienta Policial Predictiva para el pronóstico de riesgo de delitos basada en el análisis estadístico de los patrones de criminalidad espacio-temporal, y un modelo de optimización para la definición de la configuración sectores de patrullaje, adaptado a los requisitos del CNP. Dado que el modelo es necesario para ser incluido en el SSD, se espera que sea interactivo. Por lo tanto, hemos implementado un algoritmo heurístico que proporciona soluciones aceptables rápidamente. Al combinar el algoritmo propuesto con un modelo de predicción de riesgo de crimen [80, 97], se obtiene un sistema de patrullaje predictivo. Para El CNP, la implementación de un sistema de patrullaje predictivo también representa un cambio de paradigma, desde la detención hasta la prevención, lo que resulta en reducción en el coste de detención y una mejora en el nivel real, subjetivo y social de la seguridad.

El resto de esta tesis se organiza de la siguiente manera. En el Capítulo 2 se realiza una revisión de la literatura sobre el PDDP, se presenta el paradigma de la Policía Predictiva, y analiza el desarrollo y uso de los nuevos avances tecnológicos, incluyendo SSD, en el CNP. El problema de la división de un territorio representado

como una matriz en sectores convexos homogéneos se desarrolla en el Capítulo 3. En el Capítulo 4 el modelo de optimización se integra en un SSD integral para la aplicación de un paradigma de patrullaje inteligente. En el Capítulo 5 se extiende el modelo original para ser resuelto en un grafo genérico y se proponen y comparan algoritmos de solución más eficientes. Los algoritmos propuestos son probados en un estudio de caso real en el Capítulo 6. Por último, la tesis se concluye con el Capítulo 7 que presenta un resumen de la contribución principal de este trabajo, así como algunas pautas para futuras investigaciones.

CHAPTER 2

LITERATURE REVIEW

This chapter first presents the problem of defining districts and contextualizes it in the framework of police resource allocation. Second, a conceptual classification of previous works according to attributes considered and methodologies adopted is presented and insights are provided. Also, the development and use context of new technological advances in the Spanish National Police Corps is described.

2.1. The Districting Problem

District design can be seen as the problem of grouping elementary units (or atoms) of a given territory into larger districts (or clusters), according to relevant attributes (or criteria). Depending on the problem faced, the attributes considered might belong to different contexts, including economic, demographic, geographic, etc. In the last decades, the districting problem has been applied to a broad number of fields, including:

- Electric power districting [2, 3].
- Emergency service districting [51, 62].
- Internet networking [78].
- Health information systems [10].
- Police patrol districting.
- Political districting for the definition of electoral areas [9, 27, 68].
- Public transportation network districting [101, 102].
- Sales and service districting [7, 37].

- School districting [20, 92].
- Social facilities districting [70].
- Solid waste disposal districting [44].
- Winter service districting [73, 74].

A unified territorial design model that allows the formulation and solution of districting problems in a variety of applications is the subject of Kalcsics and Schröeder [53]. The authors also review the existing literature in territorial design, highlighting application fields, criteria, and solution methodologies for solving these types of problems.

2.2. The Police Districting Problem

In the United States, police departments partition the territory under their jurisdiction according to a hierarchical structure: command districts (or precincts), patrol sectors (or beats), and reporting districts (or r-districts). Each command district hosts its headquarters where the commanding officer supervises the operations. A command district is subdivided into patrol sectors, each having at least one car assigned to patrol the area and attend to the calls originating from it. Finally, r-districts constitute the atomic element in the hierarchy: the smallest geographical unit for which statistics are kept. As reported by Sarac et al. [89], r-districts can coincide with census block groups. In Europe, the territorial organizational structure of police departments depends on the country or the region considered. Nevertheless, a hierarchal structure similar to the one adopted in the United States is predominant.

The PDP concerns the optimal grouping of r-districts into externally "homogeneous" patrol sectors. In fact, the car assigned to the patrol sector should attend to all the incidents taking place in the area. Normally, if the car is busy responding to a call when another incident happens, a car from a neighboring area has to attend to it. As Mayer [67] points out, this generally leads to a domino effect, where cars are pulled from their area to another, leaving the patrol sector unattended and, therefore, more susceptible to criminal incidents. In the light of this scenario, a balanced workload among the districts and the enforcement of a maximal response time become of primary importance.

The first paper on the PDP is presented by Mitchell [71], which proposes a clustering heuristic for the redesign of patrol beats in Anaheim, California. The author considers the total expected weighted distance to incidents, as well as a workload measure defined as the sum of the expected service time and the expected travel time. Bodily [8] adopts a utility theory model that incorporates the preferences of three interest groups, namely, the citizens, the administrators, and the service personnel. A simple local search algorithm swaps patrol beats from one sector to another to improve the value of the utility function. Benveniste [1] was the first author to include workload equalization in the optimization process, solving a non-linear stochastic model by means of an approximation algorithm. D'Amico et al. [31] solve a police districting problem subject to constraints of contiguity, compactness, convexity, and equal size. The novelty of the model lies in the incorporation of queuing measures to compute patrol office workloads and response times to calls for service, computed by external software, PCAM [21, 22]. PCAM optimizes a queuing model for the deployment of police resources, providing the optimal number of cars per district. The authors solve the problem by means of a simulated annealing algorithm that iteratively calls the PCAM routine. At each step, the neighborhood is determined by a simple exchange procedure that takes into account the following constraints: the average response time per district is bounded from above; the ratio of the size of the largest and smallest districts is bounded from above; districts must be connected; the ratio of the longest Euclidean path and the square root of the area in each district is bounded from above to preserve compactness; districts must be convex. The algorithm is applied to a real-world case for the Buffalo Police Department, NY. The following objectives were considered: minimize the maximum workload (by decremental bounding constraining) and minimize the maximum average response time. A different approach is proposed by Curtin et al. [29], who apply a covering model to determine the police patrol sectors. The covering model defines the centers of the police patrol sectors in such a way that the maximum number of (weighted) incidents is covered. An incident is considered to be covered if it lies within an acceptable service distance from the center of a patrol sector. The model is integrated in a GIS and applied to a case study involving the City of Dallas, TX. In a subsequent article, Curtin et al. [30] extend their covering model to include backup coverage (e.g., multiple coverage of high priority locations). The resulting model is bi-objective in nature. The authors propose a single objective

model that maximizes the priority weighted coverage (i.e., a location is counted separately each time it is covered), while ensuring a minimum covering level in terms of the priority-weighted number of locations covered (each covered location counted only once). The model is tested on the Dallas data and refinements of the model are proposed (e.g., maximum workload per patrol sector). Zhang and Brown [114] propose a parametrized redistricting procedure for police patrols. The methodology consists of a heuristic algorithm that generates alternative districting plans. Next, the plans are evaluated in terms of the average response time and workload. With this aim, an agent-based simulation model was implemented in a GIS. The location and times of the incidents taking place in each district were modeled by an empirical distribution based on real incident data. Finally, the procedure identifies the set of non-dominated solutions. The methodology has been tested on a case study based on the Charlottesville Police Department, VA. An extensive annotated bibliography (Subsection 2.3) provides insights and detailed information on the works presented in the literature.

2.2.1. Attributes

While analyzing the existing literature on the PDP, certain basic features common to all the contributions could be identified. In fact, all the applications considered include measures for workload, response time, and the geometrical properties of the districts. Nevertheless, the implementations vary considerably. Unlike Kalcsics and Schröeder [53], the term "attributes" has been adopted instead of "criteria", with the aim of providing a more generic framework that classifies all the relevant characteristics of a PDP solution, regardless of whether they are optimized in the objective function, or expressed as constraints.

Workload Given the complex nature of police procedures and operations, and the great variability of the tasks that an agent can undertake, defining the workload could be complicated. Bruce [14] provided a set of questions that can help clarifying what needs to be considered as part of workload. Albeit difficult, an accurate definition of workload is desirable, as it ensures homogeneity in terms of the quality and speed of service, and equalizes the burden on police officers [8].

In the literature on the PDP, different definitions of workload have been adopted. In Mitchell [71], the workload is computed as the sum of the total

expected service time and the total expected travel time. Curtin et al. [29, 30] use the number (or frequency) of calls (or incidents) occurring at each district as a proxy for the workload. As different calls can have different service times, some authors consider this measure to be too naïve, as it might produce unbalanced patrol districts. In Bodily [8] and D'Amico et al. [31], workload is defined as the fraction of working time that an agent spends attending to calls. An equivalent measure is proposed by Benveniste [1]. Given the stochastic nature of her model, workload is measured in terms of the probability of a patrol car being busy. Once the probability is known, the time spent attending and answering calls can be easily calculated. More recently, workload has been defined as a combination of different characteristics. In Sarac et al. [89], the authors aggregate population and call volume. Kistler [59] makes use of the convex combination of total hours worked (i.e., from dispatch to call clearance), number of calls, and population. Finally, Zhang and Brown [114] consider both the average travel time and the response time.

Response Time Response time is an important performance measure: it is the time between the arrival of a call for service and the arrival of a unit at the location of the incident. According to Bodily [8], a reduction in the response time results in a number of beneficial effects, such as:

- Increased likelihood of intercepting a crime in progress.
- Deterrent effect on criminals.
- Increased confidence in the police.

Generally speaking, most authors only take into consideration the travel times [8, 59, 71, 114] or travel distances [1, 29, 30]. The only study considering the queuing effect is D'Amico et al. [31], where the authors apply an external model, PCAM [21, 22], to compute the total response time including the queuing time of the calls and the travel time to the location of the incident.

Geometry In 1812, Albright Gerry, the Governor of the Commonwealth of Massachusetts at the time, manipulated the division of his state and proposed a salamander-shaped district to gain an electoral advantage, leading to the expression "gerrymandering" (resulting from merging "Gerry" and "salamander"). Since then, designing electoral districts having certain geometric properties has been of

primary importance to ensure neutrality and prevent political interference in the districting process.

In the context of the PDP, geometric attributes are still relevant for efficiency (e.g., establishing boundaries that would be easy to patrol and would not frustrate the patrol officers) and for administrative reasons (e.g., coordinating with other agencies). To the best of our knowledge, only three works have explicitly included the geometric properties in the design process, such as the properties of compactness [31, 59, 89], contiguity [31, 89], and convexity [31], which are generally obtained as a consequence of optimizing the travel distance or the travel time. Also, the district area is considered in all the mentioned works. Additionally, in Kistler [59], the total length of the streets in a district is included.

Other Attributes Recently, a number of attributes that do not fall into any of the previous categories have been introduced. These attributes generally try to capture complex real-world requirements. The Buffalo Police Department needed to redesign the r-districts in such a way that the existing district boundaries would be respected, and the access to demographic data as well as their use by other agencies would be easy [89]. The Tucson Police Department needed to consider the boundaries of gang territories, city council wards, neighborhood associations, and the Davis–Monthan Air Force Base [59]. Finally, in Curtin et al. [30], backup coverage (i.e., multiple coverage) of incident locations is introduced.

2.2.2. Methodologies and Approaches

Many districting approaches have appeared in the literature. In this subsection, the contributions are categorized according to the methodology adopted, and their main characteristics are presented.

Optimization Models According to Kalcsics and Schröeder [53], the first mathematical program for the Districting Problem was proposed by Hess et al. [47], and considered a neutral definition of the political districts. Since then, a large number of models have been proposed, mostly in the context of Location Analysis. Similarly, in Curtin et al. [29, 30], maximal covering models are proposed. On the other hand, Mitchell [71] presents a Set Partitioning model that considers minimizing the expected distance inside of each subset and equalizing the workload of all the

subsets. A different perspective is adopted by Benveniste [1] and D'Amico et al. [31], where patrol cars and agents are modeled as servers in a stochastic model. Benveniste [1] proposes a Stochastic Optimization model, while D'Amico et al. [31] include a queuing model inside of a simulated annealing algorithm to compute response times that incorporate queuing effects.

Geographic Information Systems (GIS) Kistler [59] uses a GIS to redesign the Tucson Police Department districts. Most commercial GIS can be extended to integrate optimization routines. In Curtin et al. [29, 30], GIS are used in conjunction with a maximal covering model. Wang [105] presents the main application areas of GIS in police practice. Among the various applications, Wang mentions the possibility of using GIS as a police force planning tool. Namely, he refers to hotspot policing and police districting. Concerning the latter, Wang identifies three main objectives: meeting a response time threshold, minimizing the cost of operations, and balancing workload across districts. The author mentions that future research in this area should explore other goals, such as minimizing the total cost (response time), minimizing the number of districts (dispatch centers), maximizing equal accessibility, or a combination of several goals. Finally, Zhang and Brown [114] implement an agent-based simulation inside a GIS.

Other Methods Two studies have adopted approaches that do not fall into any of the other categories. Bodily [8] devises a decision model based on utility theory to achieve the best solution in terms of the surrogate utility of three interest groups. The work by Sarac et al. [89] is an example of the proverbial expression "simpler is better." After attempting to redesign r-districts by using a multi-criteria set partitioning model, the authors realized that census blocks satisfied all the requirements. It is important to notice that their approach is successful because of the specific requirements the Buffalo PD imposed on the r-district configuration (e.g., easy access to demographic data, suitable for use by other agencies).

2.2.3. Local Search Methods for the Multi-Criteria Police Districting Problem

As explained by Sarac et al. [89], the use of a structure based on census districts is desirable as it allows easy access to demographic data and, at the same

time, it is suitable for use by other agencies. In terms of solution methodologies, we propose three local search algorithms for the MC-PDP on a graph, including a Tabu Search (TS). Thanks to its ability to escape from local optima and its versatility, the TS has been successfully applied to a very wide breadth of contexts and problems, such as parameter optimization [46], vehicle routing [113], hardware/software partitioning [66], and job shop scheduling [112]. We test the proposed algorithms extensively on a real dataset based on a case study of the Central District of Madrid and compare and analyze their performance statistically. Finally, we illustrate the best solutions found by the algorithms and draw operational insights.

2.3. Annotated Bibliography on the Police Districting Problem

In the following, an annotated bibliography providing a description of the most salient points, achievements, and characteristics of researches on the PDP is given. The summaries are presented in chronological order. Next, Table 2.1 summarizes the most relevant aspects of the articles included in the bibliography that propose a computational model for the PDP.

Mitchell [71]

In his seminal work, Mitchell presents a mathematical formulation for the problem of designing police patrol sectors. The model is based on the assumption that incident distribution is known and that each call is serviced by the nearest available units. Multiple incident types are considered. Each type is characterized by a service time and a vector of weights that define the importance of the incident being attended by a specified number of units. The model minimizes the total expected weighted distance. Also, the workload, defined as the sum of the expected service time and the expected travel time, is equalized across the sectors. The problem is solved by means of an adapted clustering heuristic and applied to incident data for Anaheim, California. The solution improves the sector plan adopted at the time.

Bodily [8]

Decision model based on utility theory, which makes use of the preferences of three interest groups in the design process of police sectors: citizens (minimize travel time, equalize travel time), administrators (minimize travel time, equalize travel time, and equalize workload), and service personnel (equalize workload). The problem is solved by local search algorithm that transfers one atom from one sector to another, so that the greatest improvement in terms of surrogate utility is achieved. The algorithm stops when no improvement is possible.

Benveniste [1]

The author presents a stochastic optimization problem for the combined zoning and location problem for several emergency units. Namely, the problem involves the division of an area in sub-regions, the definition of location for the servers, and a set of rules, assigning for service an alarm to a list of servers in order of preference. The objective function considered minimizes the total expected distance between the alarm and the first available server. Stochastic alarms rates, alarms spatial density functions, and probabilities that the servers are busy are considered. The resulting model is a non-linear program. The author proposes an approximation algorithm. An equal workload case is also examined and solved.

Sarac et al. [89]

The authors describe a study undertaken to reconfigure the police reporting districts used by the Buffalo PD. The following districting criteria were considered: homogeneity in terms of population, area and call volume; geometrical properties such as compactness and contiguity; feasibility with respect to existing boundaries of five police districts; easy access to demographic data for each district; suitability for use by other agencies. Initially, the authors formulated the problem as a multi-objective set partition problem which proved incapable to solve the real-world size problem at hand. Subsequently, a practical approach has been proposed: basically, the new districts were defined according to the census block groups that intrinsically present most of the desired characteristics (homogeneity in terms of population, compactness, contiguity, easy access to demographic data, and suitable for use by other agencies). With minor modifications, this solution proved to be very effective.

D'Amico et al. [31]

The authors solve a police districting problem subject to constraints of contiguity, compactness, convexity, and equal size. The novelty of the model lies in the incorporation of queuing measures to compute patrol offices workloads and response times to calls for service, computed by external software, PCAM. PCAM optimizes a queuing model for deployment of police resources, providing the optimal number of cars per district. The authors solve the problem by means of a simulated annealing algorithm that iteratively calls the PCAM routine. At each step, the neighborhood is determined by a simple exchange procedure that takes into account the following constraints: the average response time per district is bounded from above; the ratio of the size of the largest and smallest districts is bounded from above; districts must be connected; the ratio of the longest Euclidean path and the square root of the area in each district is bounded from above to preserve compactness; districts must be convex. The algorithm is applied to a real-world case for the Buffalo Police Department. The following objectives were considered: minimize the maximum workload (by decremental bounding constraining) and minimize the maximum average response time.

Curtin et al. [29]

The authors apply a covering model to determine police patrol sectors. The covering model defines the centers of police patrol sectors in such a way that the maximum number of (weighted) incidents is covered. An incident is considered to be covered if it lies within an acceptable service distance from the center of a patrol sector. The model is integrated in a GIS and applied to a case study on the City of Dallas, Texas. In the final part of this chapter, the authors present a number of possible refinements to their model, including a maximum workload (in terms of number of weighted incidents) per patrol sector restriction.

Scalisi et al. [90]

The issue of Geography & Public Safety presents numerous articles by police analysts describing their experiences with police redistricting in the jurisdictions of their police department.

- Bruce [14] (C. Bruce, President of the International Association of Crime Analysts) poses some interesting questions that an analysts should answer to determine how workload should be measured.
- Kistler [59] (A. Kistler, from the Tucson Police Department) devises a districts evaluation measure built as the weighted sum of the following criteria: total hours worked, number of call responses, average response time, total length of all streets within the division, area of the division, and population. TPD staff used a GIS in combination with a software program called Geobalance to manually design alternative redistricting options. Future evaluations of the implemented plan showed that the projected workload ratios were reliable and realistic.
- Douglass [32] (J. Douglass, from the Overland Park Police Department) explains how the introduction of a new real-time deployment paradigm, based on criminal hotspots identification and treatment, had been implemented in the department. Unfortunately, no long-term statistical analysis was available at the time the article was written.
- Mayer [67] (A. Mayer, from the East Orange Police Department) reports a similar strategy. In fact, the East Orange Police Department implemented a geographical technology called Tactical Automatic Vehicle Locator (TAC-AVL). TAC-AVL allows for GPS tracking, visualization on a map, and recording of information regarding patrol cars and incidents. This tool has been coupled with a new deployment strategy that allows for the introduction of Impact, Resource, Response, and Conditions cars to backup understaffed zones of the jurisdiction.
- Mielke [69] (P. Mielke, from Redlands Police Department) explains how to use ESRI districting tool, a free extensions for ESRI ArcGIS that allows creating new police districts in a city or region.
- Other successful applications of geographical technologies to police redistricting have been reported from Austin PD and Charlotte-Mecklenburg PD.

Curtin et al. [30]

Following Curtin et al. [29], the authors extend the covering model to include backup coverage (e.g., multiple coverage of high priority locations). The

resulting model is bi-objective in nature. The authors propose a single objective model that maximizes the priority weighted coverage (each time a location is covered is accounted for separately), while ensuring a minimum covering level in terms of priority weighted number of locations covered (each covered location is accounted for only once). The model is tested with the police geography of Dallas, Texas, and refinements on the model are proposed (e.g., maximum workload per patrol sector).

Zhang and Brown [114]

In this work a parametrized redistricting procedure for police patrols is proposed. The methodology consists of a heuristic algorithm that generates alternative districting plans. Next, the plans are evaluated in terms of average response time and workload. With this aim, an agent-based simulation model was implemented in a GIS. The location and times of the incidents taking place at each district were modeled by an empirical distribution based on real incident data. Finally, the procedure identifies the set of non-dominated solutions. The methodology has been tested on a case study based on the Charlottesville Police Department, VA, USA.

Wang [105]

The author takes us on a trip across the main application areas of GIS in police practices. Among the various applications, Wang mentions the possibility of using GIS as a police force planning tool. Namely, he refers to hotspot policing and police districting. Concerning the latter, Wang identifies three main objectives: meeting a response time threshold, minimizing the cost of operation, and balancing workload across districts. The author mentions the work by Curtin et al. [29, 30] and states that future works in this area should explore other goals, such as minimizing total cost (response time), minimizing the number of districts (dispatch centers), maximizing equal accessibility, or a combination of multiple goals.

2.4. Predictive Policing

The term Predictive Policing is relatively recent and refers to the application of quantitative techniques to foretell where crimes will take place in the short-

Table 2.1. Mapping of attributes considered and methodology adopted, by article.

Reference	Attributes				Methodology
	Workload	Response Time	Geometry	Other	
Mitchell [71]	Expected service time, expected travel time	Expected travel time			Modified clustering heuristic
Bodily [8]	Fraction of time spent in servicing calls	Average travel time			Utility theory
Benveniste [1]	Probability of a server being found busy	Total expected station-alarm distance			Stochastic optimization
Sarac et al. [89]	Homogeneity in terms of population and call volume		Area, compactness, contiguity	Easy access to demographic data, suitable for use by other agencies, and respect of existing district boundaries	Redefinition according to census blocks
D'Amico et al. [31]	Utilization of servers	Queuing response time and travel time	Size, compactness, contiguity, convexity		Queuing model and simulated annealing
Curtin et al. [29]	Maximum incident load per sector	Maximum service distance			GIS and mathematical programming optimization
Kistler [59]	Total hours worked, number of calls, population	Average travel time	Area, total length of streets, compactness	Boundaries of gang territories, city council wards, neighborhood associations, and Air Force Base	GIS
Curtin et al. [30]	Maximum incident load per sector	Maximum service distance		Backup coverage	GIS and mathematical programming optimization
Zhang and Brown [114]	Homogeneity in terms of average travel time and response time	Average travel time			GIS and agent-based simulation

term future. The National Institute of Justice (NIJ) defined it as taking data from disparate sources, analyzing them, and then using the results to anticipate, prevent, and respond more effectively to future crimes [75]. This technique is based upon advances in criminology, such as Hot Spot theories [94, 108, 109], and studies of the ecology of crime [11, 15]. Statistics based methods have been used since the release of CompStat in 1994, however it was only a few years ago that complex mathematical algorithms have been developed to address this problem in the most profound way. CompStat combined Geographic Information System (GIS) and crime mapping techniques to identify areas of high crime intensity. The importance of measuring the occurrence of crimes in the police districts and keeping track of the actions of the police managers for decision-making was proved by Weisburd et al. [110]. This topic was opened to different approaches by the International Journal of Forecasting, which published a special issue on crime forecasting in 2003 [41].

Years later, researchers at UCLA started a new approach to the investigations of crime agglomerations, modeling the dynamics of crime hotspots and determining the parameter values that lead to the creation of stable hotspots [98]. In a subsequent study, they used amplitude equations to study the development of crime hotspot patterns [97] and self-exciting point processes [72]. Also, they mathematically proved that there were different types of hotspots, even though they seemed similar at first sight. This breakthrough was further developed using Lévy Flight models by Chaturapruek et al. [24]. More recently, Zipkin et al. [115] introduced a police behavior component aiming at suppressing hotspots of criminal activity. In this model, the police deployment adapts dynamically to changing crime patterns, making criminals modify, to a certain degree, their awareness and their criminal actions.

Probably the most ambitious predictive policing project so far made use of the algorithms created by Brantingham and Mohler, along with LAPD Captain Sean Malinowski. With three years of data, and focusing on three types of crime in particular (i.e., burglary, automobile theft, and theft from automobiles), the algorithm points out areas of likely crime incidence. The first analyses have shown a reduction of property offenses where this methodology has been implemented, reporting considerable reductions in serious violence crimes in the treatment cities and areas relative to comparison cities and areas. Another experiment of predictive policing was implemented in Santa Cruz, CA, where predictive maps based on risk

percentages were given to security managers. The use of these maps resulted in a 19% drop in burglaries [35].

Another line of research that has been widely applied in practice has focused on Risk Terrain Modeling (RTM) [19]. According to its creators, RTM is "an approach to risk assessment in which separate map layers representing the influence and intensity of a crime risk factor at every place throughout a geography is created in a geographic information system (GIS). Then all map layers are combined to produce a composite 'risk terrain' map with values that account for all risk factors at every place throughout the geography" [18]. RTM has also been proposed as a methodology for the identification of risk clusters and the distribution of police resources [55].

A number of models making use of methodologies other than hotspot and RTM have been presented in the academic literature. Xue and Brown [111] and Smith and Brown [99] developed a spatial choice model and represented criminal events as point processes combining discrete choice techniques and data mining. They used this approach to predict the spatial behavior of criminals, comparing it with existing hotspot analyses. Furtado et al. [36] model criminal behavior by using ant-inspired systems, trying to discover strategies for efficient police patrolling that take into account the dynamics of the criminals. Wang [105] used a spatio-temporal analysis for modeling criminal incidents, making use of a variety of data types, such as spatial, temporal, geographic, and demographic data. In a subsequent paper, Wang et al. [106] extended this prediction model to include information proceeding from social network posts. A similar approach is proposed by Gerber [38], finding that by combining historical crime records with Twitter data from users in a specific geographic area, the prediction performance improves for 19 of 25 crime types. Finally, Chen et al. [25] applied spatio-temporal analysis methods to investigate patterns of offenses against property.

Mielke [69], from Redlands Police Department, explains how to use ESRI districting tool, a free extension for ESRI ArcGIS that allows creating new police districts in a city or region. Other successful applications of geographical technologies to police redistricting have been reported from Austin PD and Charlotte-Mecklenburg PD.

2.5. DSS for Efficient Policing

The application of DSS to police environment has a growing role in the literature. Some innovative strategies have been established to be effective and efficient when deciding on the best officer deployment and schedule. In the late 1980s, Taylor and Huxley [103] proposed a scheduling system driven by shift turnovers, based on years of calls data. This led to a declining response time compared to manual methods. However, in that study, different crime records were not considered. More recently, Xue and Brown [111] proposed a DSS in which criminal events were modeled as point processes. They intended to predict the spatial behavior of active criminals. By analyzing the offenders' decisions, law enforcement would be empowered with better planning and knowledge of the spatial patterns of crime. A few years later, another intuitive method discussed by Li et al. [63] suggested a model based on a fuzzy self-organizing map, identifying the characteristic of several crime patterns, to determine a better duty deployment. However, recent research by Kuo et al. [61] argued most programs for making use of a naive before–after evaluation method. The authors developed hotspot approaches that apply GIS to combine crime rates and crash rates for prediction, aiming to reduce dispatch time. The main characteristics of the articles reviewed here are summarized in Table 2.2.

The research presented in this thesis innovates in the field of DSS for Public Security in several ways. First, most of the methodologies developed so far assume that the crime incidents can be represented as points in time and space. As we elaborate in "3.1 Data Pre-Processing Unit (DPPU)," this is not always the case, as most of the time the victims are not aware of the time or the location where the crime took place. Therefore, we developed a methodology that allows us to represent crime records having an indeterminate time or location. Second, the proposed methodology makes use of classical time series models. Although not novel, these models have two advantages: they are very good at capturing the seasonal components of the crime data and they do not require any additional spatial information (e.g., population density, average income, distance to risky locations), which makes them extremely applicable. Finally, to the best of the author knowledge, the DSS presented in this thesis is the first to combine Predictive Policing capabilities with an optimization model that explicitly provides efficient partitions of the territory into patrol sectors, rather than just a representation of

Table 2.2. References and structural characteristics of DSS for efficient policing.

Reference	Data	Objective(s)	Technology	Validated
Taylor and Huxley [103]	Provided by: San Francisco Police Department, CA. Historical calls for service, time spent by call type, percentage of calls requiring two or more police officers, and percentage of cars with two or more officers allocated.	Reduction of the cost of operations and increase of citizen safety and officer morale	San Francisco Police Department Computer Aided (CAD) System	Yes
Xue and Brown [111]	Provided by: Richmond Police Department, VA. Criminal incidents between July 1, 1997 and October 31, 1997. Includes more than 1200 crime observations.	Crime reduction	Regional Crime Analysis Program (ReCAP)	No
Li et al. [63]	Provided by: National Police Agency of Taiwan. Data originated from 20 county police bureaus in Taiwan from 2003 to 2004. Fourteen criminal categories were collected.	Crime reduction	Unknown	No
Kuo et al. [61]	Provided by: College Station Police Department, TX. Crime and crash data, from January 2005 to September 2010. Includes 65,461 offense reports and 14,712 crash reports.	Reduction of dispatching time and number of crime and traffic events	ArcGIS, KDE, Google Maps	No

the criminal hotspots.

2.6. Development and Use of New Technological Advances in the National Police

Today, the speed of technological change taking place has transformed day-to-day operations in policing as described by Roberts Roberts [84]. The adoption of technology in policing which started with the telephone has come a long way

since, and the launch of Sala 091 or police unit that enabled immediate contact between citizens and the police was a significant milestone. Utilization of the Sala 091 system increased exponentially with the arrival of mobile phones which increased many times over, the prompt response, assistance and performance of patrol cars.

Nowadays, patrol cars are in constant contact with the Sala 091 through systems of radio or 'pockets', which were once considered a technological revolution and suddenly provided the ability to reach multiple locations that needed police presence quickly, and, according to Harris, Harris [45] this transformation has changed the organization and operation of police departments significantly.

New technologies continue to emerge and continue to enable a more effective SNPC that will gradually be equipped with new and innovative systems helping them to use resources more effectively both at investigative level and crime prevention level.

Nevertheless, we cannot afford to be complacent, since it is probable that a second technological revolution will bring further substantial change to the organization and police operations [23, 45, 100], which the Spanish police should take advantage of in the interests of continuous improvement.

Let us consider the various notes and studies from the United States by different expert authors in this field, such as Groff and McEwen Groff and McEwen [43], who in their review of the use of the technology to store and retrieve information are convinced that among police forces which had implemented technology such as laptops, automated information systems field, records management systems and automated fingerprint identification systems had benefited greatly.

These advances have become the reality of the past 40 years, since new technologies have been gradually implemented as an integral part of the work of the SNPC. For example, substituting typing machines for current computers, not only accelerated the police documentation work but also facilitated effective storage and retrieval of the information generated daily by the police. Another example is the fingerprint system named Live Scan, that allows fingerprints to be taken and stored digitally with the consequent advantages shared across various facilities in different fields.

Thus, the SNPC is constantly monitoring and assessing advanced technologies for adoption by the Corps and also studying other legislations that regulate the use of such technology to ensure its correct functioning in daily operations. It

is interesting to mention that Roberts Roberts [84] in these matters, states with vehemence, the importance of technology and innovation in the daily operation of the police force. His reasoning is fairly logical as the benefits of the use of technology are evident in terms of efficiency and cost reduction. Reichert Reichert [83] agrees that technological innovation was the impulse that led to reforming of the prevention and control of crime, both by different police forces and related institutes.

In 2007 the Department of Justice of the United States, through the Bureau of Justice Statistics, carried out a study on the Law of Enforcement Management and Administrative Statistics (LEMAS). One of the advantages of using the survey of this law was its amplitude, which allowed them to generalize results and consistency. The study includes the use of computers by different police agencies, different computerized processing of police functions, the specific detail concerning budgetary expenditure distribution of officials in the various charts, technical maintenance, works, etc. Other computer tasks included the analysis of criminal investigations and the general exchange of information.

The largest police districts, serving populations greater than 250,000 inhabitants, largely used computer technology for the analysis of crime and cartography. In medium-size districts, serving populations of less than 250,000 people, computer use was essentially limited to the use of portable computers and terminals. They also reported using automated fingerprint identification systems. The majority of police departments were willing to use electronic methods for the transmission of reports, something that helped to ensure not only the transfer of information, but to do so expediently and accurately.

Use of computers within different police stations is increasing due to the progress, among other things, in own infrastructure of mobile communication and the development of ultramodern applications Roberts [84]. Different applications have become extremely accurate and simple to use. For example, computers allow sequencing of criminogenic maps in just a few minutes through the various statistical data, as opposed to what once would have taken several hours minimum to complete, as well as the different images required for distribution amongst the police to facilitate searches more easily and quickly.

Another breakthrough in terms of time and ease of data collection is the use of conjectured systems. They aggregate various criminal investigations data used by different intelligence units of police centers, and help to identify different

relational schemes that, at first, may appear different and unrelated. For example; matching suspects through incomplete names, signatures, nicknames or descriptions of physical characteristics or even vehicle related searches.

Another technological innovation in the fight against crime is the use of surveillance cameras, which enables photos and/or video to be taken of people committing various illegal acts such as painting graffiti, illegal dumping of waste, and vandalism or purely criminal acts, and notify the corresponding police agencies immediately. Some of these cameras even have the capability to warn the people who are committing illegal acts of possible prosecutions and to ask them to leave the location. Use of these types of cameras has increased effectiveness greatly, especially by becoming wireless and by using independent solar power sources [93]. Nonetheless, these types of photographic systems are not new, as traffic cameras have already been approved and in use for decades to identify traffic violations such as speeding.

Closely related to surveillance camera systems, are thermal imaging devices. These non-intrusive electronic elements are fairly easy to use, maintain and even store, and their function is to produce images of surface energy emitted or reflected. Such mechanisms offer photos of heat sources with respect to their environment, empowering the police to monitor, locate, and catch people suspected of committing criminal offences using these generators of thermal imaging. The intervening police can see the live image of body heat of the person while in pursuit, thus not preventing the police acting even if the person tries to camouflage himself through the use of dark or similar clothing.

In addition, the use of cameras by police has been developed, contributing to their proficiency. Again, let us observe data from the United States on this particular scenario, specifically the aforementioned survey LEMAS 2007, which determines that the use of cameras in different police cars was at approximately 61%, approving to be a valuable tool inside the car patrol to ensure, amongst other things, the highest level of professionalism of police during the course of their work. Videos recorded in this way in several police interventions, allowed the events to be presented as evidence if needed. This often happens in the case of citizen's complaints or even more extreme cases such as in trials against policemen.

Another possible use of this kind of medium would be purely educational, serving as training videos in different police schools. Cameras are well suited in this situation as probably the greatest strength of this technology is the ability to

act as a silent and objective witness of the different facts that develop under its watch. It can both protect and speak for the police or victims of police injustice which in many cases cannot speak for themselves. As determined by Brown Brown [12], a police officer spends approximately 92% of his time obtaining, combining and circulating the managed information. So, to access the information easily and quickly, by using it accurately and distributing it on time is a very important part of the daily police duty. Even from a more traditional approach to policing, it is necessary to access accurate data to ensure that actions taken during service and any critical decisions leading to the detention of an individual are accurate [52].

Just as importantly, traditional approaches and more sophisticated approaches regarding police action depend on even more information. The information handled may not necessarily be related to crime and its perpetrators, but often also include issues as important as top priority information of activities in the communities, serving to prepare a more collaborative approach of police action for prevention and response to future crime and its prevention. The forces of law and order have invested more than one decade in different strategies and techniques for predicting police activities, acting in specific areas with greater possibility of crimes. For example, ComStat4, is a police action oriented towards the specific problems of the geographical area, depending upon the management and analysis of information [13, 80]. Related to the technological use of lasers, it is also important to mention its particular efficacy in a transcendent subject such as terrorism. Thanks to the use of portable devices utilizing laser spectroscopy, the detection of chemical substances can be determined in just a few seconds. These devices even have the ability to reveal the chemical composition with a certainty of 95% of for example a powder suspect [93].

On the other hand, tracking devices using Global Positioning Systems (GPS), can be attached to the vehicles of suspects if the situation requires. Such a mechanism would considerably reduce the need of follow-ups and even police chases. This technology provides the police with time and valuable information about the habits of the suspect, which allows them to establish strategic plans to then study the suspect, if required, with minimal detrimental impact on nearby areas [93]. GPS technology also facilitates a response more quickly and effectively in the areas requiring police intervention, ensuring patrol is closer to the area where they need to act.

It is important to highlight that one of the biggest advantages of the tech-

nology is that it has made things simple and quick, which means more time can be spent on planning services more efficiently and effectively. As an example of how this technology has facilitated the police to identify, among other things, vehicles owners through their license plate number, known in the Spanish Police as the computer system Atlas, in a very short time, the police are able to determine vehicle details and the identity of their owners.

If a car has been stolen or has participated in a criminal act, or the owner of the car has a criminal record, this is known immediately by the acting agent. Another example would be the common use by the police of Tablets, which are useful in briefings and for obtaining, storing and exchanging information for subsequent use by several team members, even allowing the downloading of information on a computer for storage or further analysis, and will probably allow the improvement of Predictive Policing.

CHAPTER 3

A MULTI-CRITERIA POLICE DISTRICTING PROBLEM

This chapter illustrates the PDP developed in collaboration with the SNPC. The goal of the model is to partition the territory under the jurisdiction of a district into patrol sectors in the best possible way. The criteria for evaluating the goodness of the configurations of the patrol sectors were identified after interviewing several service coordinators and a number of agents involved in public safety operations. The result is a mathematical optimization model that finds an efficient configuration in terms of prevention service and attention to calls, distributing the workload equitably between the agents.

For further information on this chapter see [17].

3.1. Model Characteristics

During the interviews with the public servants involved in public safety operations, several desirable characteristics were identified in order to find a 'good' territory partition.

- **Compact Areas:** A compact area allows better control of the territory by the agents, as travel times from one point to another within the area are minimal. Therefore, the more compact an area is, the faster agents in the corresponding area can respond to emergency calls.
- **Homogeneity in terms of workload:** Generating patrol sectors that are similar in terms of workload is quite useful for two main reasons. First, it ensures a more efficient distribution of work and, therefore, better service. Second, greater equality in the workplace increases the satisfaction of the agents.

- **Mutual support:** It is desirable that agents be able to count on the support of agents assigned to other patrol sectors in case of need and emergency.

Our model differs from those proposed so far in the literature in a number of relevant aspects. In general, our focus is on crime prevention. For this reason, the purpose of our model is to increase the effectiveness of the deterrent effect of the agents' presence on the territory, by concentrating the agents in the areas with a higher risk of crime. On the other hand, previous approaches such as D'Amico et al. [31] and Zhang and Brown [114] focus on reaction to crime incidents and aim at optimizing the response to emergency calls and, hence, to crimes that have already happened. Additionally, we present the first model for the PDP that optimizes attributes of area, crime risk, compactness, and support, simultaneously.

Specifically, mutual support is an attribute that has not been included in any previous model. Mutual support differs from backup coverage [30] in that the former regards the possibility of receiving backing in any point of the patrol sector from any other agent in the district, while the latter only concerns the overlapping areas between patrol sectors. Furthermore, our model allows the decision-maker to explicitly and easily include his/her preferences in the optimization process by means of weights associated to the attributes. In the formulation proposed by D'Amico et al. [31], the user can specify his/her preferences only by adjusting the righthand side coefficients in the constraints, while in Curtin et al. [29, 30]no user preference is considered.

Finally, all the approaches previously presented in the literature require specific data and information, such as the time, location, and service time of incidents and emergency calls, which might not be available. This requirement makes these models inapplicable in any context where this information is not available. Also, these methodologies do not take into consideration, and hence they cannot be extended to, all the non-violent crimes that are not reported by emergency calls, such as, pickpockets, theft of vehicles, or property damage.

3.2. Input Data

Without loss of generality, the territory under the jurisdiction of the district is assumed to be divided into a square grid, G, of n rows and m columns, having elements indexed by $(i, j) \in G$. Following this structure, we define two data matrices both having n rows and m columns:

- **The crime risk matrix, R.** Its entries, $r_{ij} \in R$, are non-negative real numbers specifying the crime risk associated with the corresponding locations. The risk of criminal activity can be estimated with past data or using a predictive policing model [80].
- **The area matrix, A.** Its entries, $a_{ij} \in A$, are non-negative real numbers specifying the total street length at each tile of the grid. This data can be easily obtained using a GIS.

Finally, the number of patrol sectors, p, is required. The model uses this information to define the number of areas into which to partition the territory.

3.3. Notes on Taxicab Geometry

The representation of the territory as a grid necessarily involves certain simplifications when considering geometric properties such as continuity and distance. Given the loss of information on the urban fabric of streets and roads resulting from using a grid as a model, it is natural and necessary to apply a taxicab geometry. In this geometry, the distance between two points, also called the Manhattan distance, is the sum of the (absolute) differences of their coordinates. Therefore, the distance between the points $a = (i, j)$ and $b = (k, l)$ is calculated as

$$dist\,(a, b) = |i - k| + |j - l|\,. \tag{3.1}$$

Following this definition, two points are considered adjacent if and only if their distance is equal to 1. A subset of points s is defined to be connected if between any pair of points (belonging to s) there is a path of adjacent points (belonging to s) connecting them. Within a connected subset s, the minimum distance between any pair of points is defined as the length of the shortest path connecting them formed by points belonging to s. If this path does not exist, then the subset is not connected. The matrix of the shortest paths between pairs of points belonging to s, F^s, can be calculated efficiently using the Floyd–Warshall algorithm [34, 107]. We refer to its elements as $F^s_{a,b}$, where $a, b \in s$; $F^s_{a,b} = \infty$ when there is no path connecting points a and b. The connectivity condition can be expressed as

$$0 \leq F^s_{a,b} < \infty,\ \forall a, b \in s \iff s \text{ is connected.} \tag{3.2}$$

Finally, we present the property of convexity. In taxicab geometry, the definition of convexity is related to the notion of the orthogonal convex hull of a subset. In this thesis, we exploit the following property: a subset of points s is convex if, and only if, for all pairs of points belonging to s, the shortest path distance (inside of the subset) is equal to the Manhattan distance between them:

$$F^{s}_{a,b} = dist\,(a,b)\,,\ \forall a,b \in s \iff s \text{ is convex.} \tag{3.3}$$

3.4. Constraints

We now present the model constraints. As explained in previous sections, the model must generate a patrol sector configuration. The districts cannot overlap and they must cover the whole territory.

Mathematically, a partition is a family of non-empty subsets completely covering the initial set and in which each pair of these subsets are disjoint. Thus, the first condition that any solution has to satisfy is to define a partition, P, of the territory considered. This translates to a definition of the subsets over the matrices A and R. Each subset $s \in P$ contains some of the matrix entries and represents a patrol sector. From now on, the terms subset and (patrol) sector will refer to the same concept.

The second restriction concerns the cardinality of the partition. The number of subsets in the partition must be exactly p.

The third condition regards the subsets' geometry. Only connected subsets are feasible. This condition implies that an agent cannot be assigned to a patrol district composed of two or more separate areas of the city. Furthermore, all the subsets are required to be convex. When a subset is convex, it is also optimally efficient in terms of distances between its points. In fact, in a convex subset, there is a minimal shortest path connecting any pair of points. Therefore, this condition allows the generation of patrol sectors that are more efficient in terms of movement within the area.

The resulting PDP can be characterized by the following mathematical program, adopted from King et al. [57].

$$\text{opt} \qquad obj\,(P) \tag{3.4}$$

$$\text{s.t.} \quad \exists s \in P | \, (i,j) \in s \quad \forall \, (i,j) \in G \tag{3.5}$$

$$Empty_s \, (P) = 0 \quad \forall s \in P \tag{3.6}$$

$$|P| = p \tag{3.7}$$

$$Conn_s \, (P) = 1 \quad \forall s \in P \tag{3.8}$$

$$Conv_s \, (P) = 1 \quad \forall s \in P \tag{3.9}$$

In the model, $obj \, (P)$ in Equation (3.4) is an objective reflecting the goals of the decision maker. The constraints (3.5) require that all the points of the grid must belong to a subset. $Empty_s \, (P)$ in Constraints (3.6) is an indicator function that equals 1 when s is empty (i.e., no points have been assigned to it) and zero otherwise. The cardinality constraint (3.7) forces the number of subsets to be exactly p. Finally, $Conn_s \, (P)$ in Constraints (3.8) is an indicator function that equals 1 when s is connected and zero otherwise, and $Conv_s \, (P)$ in Constraints (3.9) is an indicator function that equals 1 when s is convex and zero otherwise.

3.5. Attributes

To find the best possible partition, a methodology is needed that allows the comparison of the different solutions in terms of "goodness". To evaluate this, we need to define some unambiguous criteria. More specifically, we consider the following attributes for each subset $s \in P$:

- **Area**, a^s. This attribute identifies the size of the territory that an agent should patrol. It is calculated as

$$a^s = \sum_{(i,j) \in s} a_{ij}. \tag{3.10}$$

- **Support received**, b^s. Two districts support each other if the distance between their geometric medians is less than or equal to a defined constant, K. We recommend defining K as

$$K = \left\lceil \frac{\max \{m, n\}}{\sqrt{p}} \right\rceil. \tag{3.11}$$

The geometric median, o^s, of a subset s is the point minimizing the sum of

the distances to the elements of the subset:

$$o^s = \arg\min_{a \in \mathrm{s}} \left\{ \sum_{b \in s} F^s_{a,b} \right\}. \tag{3.12}$$

Finally, the support received by a subset can be calculated as follows:

$$b^s = \left| \left\{ s' \in P \left| dist\left(o^s, o^{s'}\right) \leq K,\ s \neq s' \right\} \right|. \tag{3.13}$$

- **Demand**, c^s. The demand is defined as the total risk of the subset, i.e., the sum of the risks associated to the points belonging to the subset:

$$c^s = \sum_{(i,j) \in s} r_{ij}. \tag{3.14}$$

  It is important to remember that r_{ij} identify the crime risk associated to a point. Therefore, the demand c^s identifies how "dangerous" the subset is in terms of the expected crime risk.

- **Diameter**, d^s. The diameter of a subset is defined as the maximum distance between any pair of points belonging to the subset:

$$d^s = \max_{a,b \in s} \left\{ F^s_{a,b} \right\}. \tag{3.15}$$

  The diameter is an efficiency measure. In fact, compact districts have small diameters. Moreover, the diameter can be interpreted as the maximum distance that the agent associated to the district should travel in case of an emergency call. Therefore, a small diameter results in a low response time.

The attributes defined are not comparable, as they are associated to different dimensions. To make comparisons between them, we need to convert the attributes into dimensionless ratios:

- **Area ratio**, α^s. This is the ratio of the subset area to the whole area:

$$\alpha^s = \frac{a^s}{\sum_{(i,j) \in G} a_{ij}}. \tag{3.16}$$

- **Isolation ratio**, β^s. To express all the ratios as quantities to be minimized, we consider the isolation of a subset as the complement of the support re-

ceived:

$$\beta^s = \frac{p - 1 - b^s}{p - 1}. \tag{3.17}$$

- **Demand ratio**, γ^s. This is the ratio of the subset demand to the whole demand:

$$\gamma^s = \frac{c^s}{\sum_{(i,j)\in G} r_{ij}}. \tag{3.18}$$

- **Diameter ratio**, δ^s. This is the ratio of the subset diameter to the maximum diameter possible. We estimate this quantity as the maximum Manhattan distance between two points in the grid:

$$\delta^s = \frac{d^s}{\max_{a,b\in G}\{dist\,(a,b)\}}. \tag{3.19}$$

Now that all the attributes have been expressed in a dimensionless fashion, it is necessary to define the relative importance of each ratio. The decision maker can express preferences by associating weights to the attributes: w_α, w_β, w_γ, and w_δ. A larger weight assigns more importance to the minimization of the attribute. We can now define the workload W^s of a subset s as the sum of the products of weights with the ratios:

$$W^s = w_\alpha \cdot \alpha^s + w_\beta \cdot \beta^s + w_\gamma \cdot \gamma^s + w_\delta \cdot \delta^s. \tag{3.20}$$

3.6. Objective Function

After analyzing the information provided by the professionals of the SNPC, we identified two primary necessities that our model should take into account:

- The model should define districts that are as efficient as possible, in terms of the attributes considered and the weights specified.
- The model should define districts that are as homogeneous as possible, in terms of the attributes considered and the weights specified.

Unfortunately, there might be a trade-off between these requirements. As an example, an increase in the homogeneity of the districts could reduce the global efficiency, and vice-versa. Therefore, we define a multi-criteria objective function

that takes into consideration the preferences of the decision maker with respect to these factors:

$$\min \; obj(P) = \lambda \cdot \max_{s \in P} \{W^s\} + (1 - \lambda) \cdot \frac{\sum_{s \in P} W^s}{p}, \tag{3.21}$$

where $0 \leq \lambda \leq 1$. The term $\max_{s \in P}\{W^s\}$ represents the worst workload, while the term $\frac{\sum_{s \in P} W^s}{p}$ is the average workload[1]. The objective function defined, inspired by the extended goal programming paradigm introduced by Romero [85, 86], allows the decision maker to examine the trade-off between optimization and balance by a parametric analysis. In fact, by varying λ, the model gives a range from optimization ($\lambda = 0$) to balance ($\lambda = 1$).

3.7. The Optimization Algorithm

The model resulting from (3.4)–(3.9) is extremely complex. In fact, Drexl and Haase [33] showed that subset contiguity can be enforced by using a number of inequalities, similar to the sub-tour elimination constraints in vehicle routing, that increases exponentially with the number of subsets, making it intractable in large problems. Shirabe [95, 96] proposed a fluid flow approach to contiguity, yielding a mixed-integer program formulation that avoids this exponential increase by adding continuous decision variables measuring a flow volume. Nevertheless, this formulation is also intractable in large problems. Also, to the best of the author's knowledge, no linear formulation for the convexity condition has been presented in the literature. Additionally, for the purposes of this research, computational time is critical since the model is to be included in an integrated DSS and, therefore, the user would expect a solution within a reasonable time. Therefore, the presented model is solved by means of a heuristic algorithm. Namely, we adopt a random search algorithm that, on each iteration, generates a new solution using a randomized greedy heuristic and then improves it using a local search algorithm. Additionally, the random search algorithm can be initialized by a solution provided by the user, which is then optimized by means of local search.

Random Greedy Algorithm

[1] The average workload term of the objective function includes constant terms, such as $\sum \alpha^s = 1$ and $\sum \gamma^s = 1$. We decided to include them so that the worst workload and the average workload could have the same magnitude and, therefore, be comparable.

Algorithm 1 Random greedy algorithm.

```
procedure GreedyHeuristic(A, R, p)
                                          ▷ Phase 1 – Random initialization of the subsets.
C ← (i, j) ∈ G;                           ▷ Initialize points.
for all s ∈ P do
    c ← rand(C);                          ▷ Randomly choose a point from C.
    C ← C\c;                              ▷ Remove c from C.
    s ← c;                                ▷ Assign c to s.
end for
                                          ▷ Phase 2 – Subset expansion.
while C ≠ ∅ do
    P* ← ∅;
    for all {s ∈ P} and {c|c ∈ Neighborhood(s) ∧ c ∈ C} do
        s ← s ∪ c;                        ▷ Assign c to s.
        if Conv_s(P) = 1 and obj(P) < obj(P*) then
            P* ← P;                       ▷ Save the best solution found so far.
            c* ← c;                       ▷ Save the last point added to a subset.
        end if
        s ← s\c;                          ▷ Remove c from s.
    end for
    P ← P*;                               ▷ Update the current solution with the best solution found so far.
    C ← C\c*;                             ▷ Remove c* from C.
end while
return P*;
end procedure
```

This algorithm generates an initial solution by randomly choosing the first element of each subset and then expanding the subsets in a greedy fashion while preserving their connectivity and convexity. Initially, the partition subsets are empty. In the first phase of the algorithm, each subset is initialized with a randomly chosen point. At each iteration of the second phase, the algorithm extends the initial solution by assigning a point to a subset. The algorithm chooses the combination of point and subset that results in the best feasible solution. The algorithm ends when all the points have been assigned to subsets. However, due to the convexity condition, it is possible that the algorithm cannot assign all the points to subsets. In this case, the algorithm returns an empty set.

The procedure $rand\,()$ randomly chooses an element from the input set. The set $Neighborhood\,(s)$ returns the neighboring points, i.e., the set of feasible points that do not belong to s and whose distance from at least one of the points in s is exactly 1:

$$Neighborhood\,(s) = \{a = (i,j) \in G \backslash s \,|\exists b \in s \,|dist\,(a,b) = 1\,\}\,. \tag{3.22}$$

The neighboring set of points can be efficiently calculated by keeping a list for each

Algorithm 2 Local search algorithm.

procedure $LocalSearch(A, R, p, P)$
$improved \leftarrow$ **true**;
while improved **do**
  $improved \leftarrow$ **false**;
  $P^{\star} \leftarrow P$; ▷ Initialize the best solution found with the current one.
  for all $\{s^A \in P\}$ and $\{c \in s^A\}$ and $\{s^B \in P | c \in Neighborhood(s^B)\}$ **do**
    $s^A \leftarrow s^A \backslash c$; ▷ Remove c from s^A.
    $s^B \leftarrow s^B \cup c$; ▷ Assign c to s^B.
    if $\forall s \in P,\ Empty_s(P) = 0$ and $Conn_s(P) = 1$ and $Conv_s(P) = 1$ and $obj(P) < obj(P^{\star})$ **then**
      $P^{\star} \leftarrow P$; ▷ Save the best solution found so far.
      $improved \leftarrow$ **true**; ▷ The solution improved.
    end if
    $s^B \leftarrow s^B \backslash c$; ▷ Remove c from s^B.
    $s^A \leftarrow s^A \cup c$; ▷ Assign c to s^A.
  end for
  $P \leftarrow P^{\star}$; ▷ Update the current solution with the best solution found so far.
end while
return $P^{\star}$;
end procedure

subset that is updated every time a point is added to or removed from the subset. Subsets can be checked for convexity ($Conv_s(P) = 1$) by applying condition (3.3), having a complexity equal to $O\left(|s|^2\right)$. King et al. [57, 58] propose data structures specifically designed for the efficient implementation of contiguity and hole constraints in local search algorithms for planar graph partitioning. Nevertheless, implementing such sophisticated data structures in our algorithm is unnecessary, as no real benefit would result from reducing the complexity of the convexity test. In fact, the complexity for running the convexity test is dominated by that of the Floyd–Warshall algorithm $O\left(|s|^3\right)$ to compute the shortest-path distance matrix, on which the convexity test is based.

Local Search Algorithm

The local search algorithm improves the solution generated by the greedy algorithm by reassigning the points located at the subsets' borders. At each step of the algorithm, all the feasible reassignments of a point are considered. The algorithm chooses the reassignment that results in the best partition. If the solution found is better than the previous one, then it is taken as the starting point of the next iteration.

Subsets can be checked for connectivity ($Conn_s(P) = 1$) by applying condition (3.2), having a complexity equal to $O\left(|s|^2\right)$. Also the connectivity test requires the shortest-path distance matrix computed using the Floyd–Warshall algorithm.

Algorithm 3 Random search algorithm.

```
procedure RandomSearch(A, R, p, N, P̂)
if P̂ ≠ ∅ then
    P* ← LocalSearch(A, R, p, P̂);          ▷ Initialization by user provided solution.
else
    P* ← ∅;                                 ▷ Initialize the best solution found to empty set.
end if
n ← 0;                                      ▷ Initialize the number of iterations to zero.
while Loop() do
    P ← GreedyHeuristic(A, R, p);           ▷ Generate a new solution.
    P ← LocalSearch(A, R, p, P);            ▷ Improve the current solution.
    if obj(P) < obj(P*) then
        P* ← P;                             ▷ Save the best solution found so far.
    end if
    n ← n + 1;                              ▷ Increase the iteration counter.
end while
return P*;
end procedure
```

Random Search Algorithm

Initially, if no initial solution $\hat{P}$ is provided by the user, the best solution is initialized to empty. Otherwise, the best solution is initialized by optimizing $\hat{P}$ by means of local search. At each iteration, the random search algorithm generates a new solution by calling *GreedyHeuristic* and *LocalSearch*. The new solution is compared with the best solution found. The algorithm iterates according to a certain looping condition, *Loop*. In our implementation, the algorithm runs for a fixed amount of computational time.

CHAPTER 4

A DECISION SUPPORT SYSTEM FOR PREDICTIVE POLICE PATROLLING (P^3-DSS)

A Decision Support System (DSS) can help to optimize effective use of the scarce human resources available. In this thesis we present a DSS that merges Predictive Policing capabilities with a Patrolling Districting Model for the design of predictive patrolling areas. The proposed DSS, developed in close collaboration with the Spanish National Police Corps (SNPC), defines partitions of the territory under the jurisdiction of a district that are efficient and balanced at the same time, according to the preferences of a decision maker. To analyze the crime records provided by the SNPC, a methodology for the description of spatially and temporally indeterminate crime events has been developed.

For further information on this chapter see [16].

4.1. Structure of the P^3-DSS

The DSS we propose is composed of three main elements that identify the predictive police patrolling strategies: Data Pre-Processing Unit (DPPU), Crime Risk Forecasting Unit (CRFU), Patrol Sector Optimization Unit (PSOU).

Figure 4.1. The P^3-DSS main loop.

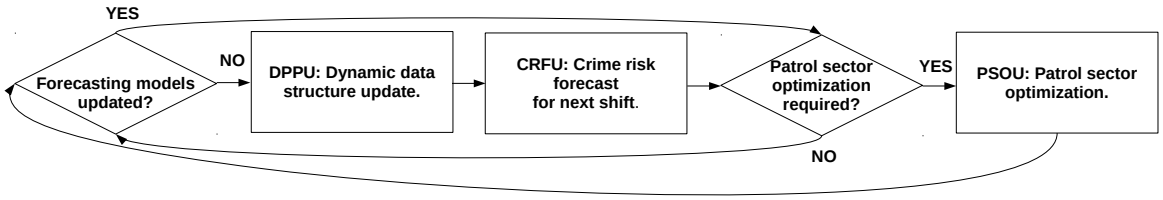


Figure 4.1 illustrates the main loop of the P^3-DSS, which shows how the elements interact. Being a real-time system, the P^3-DSS undertakes an infinite

loop composed of two main parts. When a certain updating condition is met (e.g., when a significant number of records have been added to the crime reports database and a certain amount of time has passed since the last update), the system first calls the DPPU to update the internal data structures and the CRFU to update the forecasting models. These operations are carried out in the background, and are invisible to the user. Whenever a request for a patrolling configuration is sent by the user, the P^3-DSS calls the PSOU and returns the resulting configuration. These units will be presented in detail in the following subsections.

4.1.1. Data Pre-Processing Unit (DPPU)

Most of the research in Predictive Policing assumes that the criminal incidents are associated with a determined point in time and space. Ratcliffe [82] presents a methodology for the temporal description of crime events where the time of incidence is indeterminate. However, many common crimes also have indeterminate spatial incidence, e.g., pickpocketing. We now present a novel methodology for the spatio-temporal description of crime events that can be indeterminate in both the temporal and the spatial dimensions.

The P^3-DSS makes use of a three-dimensional data structure providing a discretized representation of the space (i.e., the territory under the jurisdiction of a district) and of time (i.e., the period of time considered in the historical data). In fact, we represent the territory under the jurisdiction of a district as a grid, G, having I rows and J columns. The size of the grid cells can be determined by taking advantage of the results of Gorr and Harries [41], which show that the average monthly crime counts in a territory need to be on the order of 30 or more to achieve good forecast accuracy. Time is discretized by considering the agents' shift as the time unit[1]. The total number of time steps, T, can be easily computed by calculating the number of shifts included in the period of time encompassed by the historical data available.

The main data structure used by the DSS is a three-dimensional array C, having dimension $I \times J \times T$. The value of each element, $c_{i,j,t} \in \mathbb{R}$, represents the number of crime reports associated with location $(i, j) \in G$ at time step $t \in \{1, \ldots, T\}$. The procedure executed by the DPPU to compute this value will be

[1] In the SNPC, shifts are scheduled as follows: morning shift, from 8 AM to 3 PM; afternoon shift, from 3 PM to 10 PM; night shift, from 10 PM to 8 AM of the next day.

explained in the following.

The array C is initialized to 0. Next, each crime report is proportionally accounted for in the elements of the array involved in the criminal event according to the following data included in the crime report database of the SNPC:

- Event time window: Range of dates and times in which an event occurred.
- Event location: Place where the crime was committed, i.e., its geographical location. This might be specified as an address or, more generally, as an area.

By "proportionally" we mean that a crime is partially accounted for in all elements $c_{i,j,t}$ referenced by the data. The following cases might occur:

- The time and location of the crime are known with certainty and are limited to a single grid cell and time step (e.g., a robbery). The location (i, j) and time step t can be determined unambiguously. The value of $c_{i,j,t}$ is increased by 1.
- The location of the crime is limited to a single grid cell and the time is expressed as a period of time covering more than one time step (e.g., a motor vehicle theft). In this case, the location (i, j) can be determined unambiguously but the time is expressed as a range $t, \ldots, t + n$. Thus, the values of the $c_{i,j,t}, \ldots, c_{i,j,t+n}$ are increased according to the proportional part of the crime time range that falls into each time step.
- The time of occurrence of the crime is contained in one time step but the location is not limited to a single grid cell (e.g., a breach of the peace). The time step t can be determined unequivocally but the location is expressed as set of locations $\{(i_1, j_1), \ldots, (i_n, j_n)\}$. Therefore, the $c_{i_1,j_1,t}, \ldots, c_{i_n,j_n,t}$ are increased according to the proportional part of the area considered by the reports that falls into each grid cell.
- The location is not limited to a single grid cell and the time of the crime is contained in more than one time step (e.g., a pickpocketing or evading a police car). In this case, first all the elements involved are identified, and then the value of each element is increased proportionally, as illustrated in the previous items.

Once C is built, we can use it to forecast the risk of crime in a specific shift.

4.1.2. Crime Risk Forecasting Unit (CRFU)

The array C can be looked at from two different perspectives. In fact, by selecting a specific time step, $\bar{t} \in \{1, \ldots, T\}$, the bi-dimensional matrix $C_{\bullet,\bullet,\bar{t}}$ represents the distribution of crimes reported in the territory for the selected shift. Similarly, by selecting a specific location $(\bar{i}, \bar{j}) \in G$, the vector $C_{\bar{i},\bar{j},\bullet}$ is the time series of the crime counts for the selected location. Since this number is an approximation to the real number of crimes committed, we can apply classical time series forecasting models to predict the risk of crime for each cell of the grid. For instance, exponential smoothing models assign progressively smaller weights (importance) to older data, whereas newer data is given progressively greater weight. The use of classical time series forecasting models has also been validated by previous research on the topic. In fact, Gorr et al. [42] and Cohen [28] agree that exponential smoothing models are very accurate at forecasting crime series at the sub-district level. Following these results, the CRFU considers for each location $(i, j) \in G$ an exponential smoothing state space model [50]. As shown in Section 6.2, although this methodology relies exclusively on crime location, the CRFU is capable of discerning the underlying pattern and producing good quality predictions.

4.1.3. Patrol Sector Optimization Unit (PSOU)

Forecasting the risk of crime in an area is just the first step towards the definition of sound patrolling sectors in a district. By optimizing the configuration of patrol areas, it is possible to focus resources on the most relevant locations, with a consequential improvement in the effectiveness of patrolling operations. After interviewing several service coordinators and a number of agents involved in public safety operations, several desirable characteristics were identified in order to find a 'good' territory partition.

- **Compact Areas**
- **Homogeneity in terms of workload**
- **Mutual support**

The mathematical optimization model proposed for the solution of this problem partitions the area under the jurisdiction of a police district into a defined

number of patrolling sectors, in the most efficient way. The resulting districting problem, called the Multi-Criteria Police Districting Problem (MC-PDP), has been presented in Chapter 3 and a fast heuristic algorithm was proposed for its solution. Computational experiments showed that the MC-PDP rapidly generates patrolling configurations that are more efficient than those currently adopted by the SNPC. The main characteristics of the MC-PDP are introduced below.

Input Data and Parameters

The PSOU requires the following input data:

- Let R be the bi-dimensional crime risk matrix for a future time step $t' > T$, having dimension $I \times J$. This matrix is computed by the CRFU by forecasting the crime risk level at each location $(i, j) \in G$. Thus, the value of every element $r_{i,j} \in R$ represents the predicted risk of crime at location (i, j) and time step t'.
- Let A be the bi-dimensional distance matrix, having dimension $I \times J$. The elements $a_{i,j} \in A$ are non-negative real numbers that represent the total length of the streets to be patrolled at location $(i, j) \in G$. This matrix can be computed using the information provided in a GIS.
- Let $p \in \mathbb{N}$ be the number of patrolling sectors to be defined. We assume $p > 1$.
- Let $\mathbf{w} \in \mathbb{R}^4$ be the vector of weights expressing the decision maker's preference associated with each attribute (see paragraph "Patrol Sector Attributes and Workload" below).
- Let $\lambda \in \mathbb{R}$, $0 \leq \lambda \leq 1$ be the coefficient expressing the decision maker's preference between optimization and workload balance (see paragraph "Objective Function" in page 49).

Structure of a Patrolling Configuration

A feasible patrolling configuration is a partition P of the territory considered. Each subset $s \in P$ represents a patrol sector and is expressed as a subset of locations, i.e., $s \subseteq G$. From this point onward, the terms "patrol sector" and "partition subset" will refer to the same concept. The number of subsets in the partition must be exactly p. All partition subsets must be connected and convex.

Patrol Sector Attributes and Workload

The MC-PDP evaluates the patrol sectors $s \in P$ defined by a configuration P according to four main attributes: area, isolation, demand, and diameter. All the attributes, explained in the following, are expressed as dimensionless ratios, so as to be comparable.

- **Area**, α^s. This attribute is a measure of the size of the territory that an agent should patrol. It is expressed as the ratio of the area encompassed by sector s, to the whole district area.

$$\alpha^s = \frac{\sum_{(i,j)\in s} a_{ij}}{\sum_{(i,j)\in G} a_{ij}}. \tag{4.1}$$

- **Isolation**, β^s. In the MC-PDP, two districts support each other if the distance between their geometric medians (i.e., the location in a sector that minimizes the sum of the distances to all the locations in the sector) is less than or equal to a defined constant, K. We recommend defining K as

$$K = \left\lceil \frac{\max\{I, J\}}{\sqrt{p}} \right\rceil. \tag{4.2}$$

  The support received by a sector can be calculated by

$$b^s = \left| \left\{ s' \in P \,\middle|\, dist\left(o^s, o^{s'}\right) \leq K,\ s \neq s' \right\} \right|, \tag{4.3}$$

  where o^s identifies the median location of sector s and $dist$ is the distance between two locations[2]. The isolation of sector s is computed as

$$\beta^s = \frac{p - 1 - b^s}{p - 1}. \tag{4.4}$$

- **Risk**, γ^s. This attribute is a measure of the total risk associated to the sector that an agent patrols. It is expressed as the ratio of the total risk of sector s, to the whole district risk.

$$\gamma^s = \frac{\sum_{(i,j)\in s} r_{ij}.}{\sum_{(i,j)\in G} r_{ij}}. \tag{4.5}$$

- **Diameter**, δ^s. The diameter of a subset is defined as the maximum distance between any pair of locations belonging to that subset. It has been introduced in the MC-PDP as an efficiency measure. In fact, the diameter

[2]Given the underlying grid structure, the distance function used in the MC-PDP is the Manhattan distance.

can be interpreted as the maximum distance that the agent associated to the district would have to travel in case of an emergency call. Therefore, a small diameter results in a low response time. The diameter measure used to evaluate a patrol sector is the ratio of the subset diameter to the maximum diameter possible.

$$\delta^s = \frac{\max_{a,b \in s} \{dist\,(a,b)\}}{\max_{a,b \in G} \{dist\,(a,b)\}}. \tag{4.6}$$

By combining the attributes with the preference weights $\mathbf{w}$ defined by the decision maker, we can compute a measure of the workload W^s of a sector s as

$$W^s = w_\alpha \cdot \alpha^s + w_\beta \cdot \beta^s + w_\gamma \cdot \gamma^s + w_\delta \cdot \delta^s. \tag{4.7}$$

Objective Function

According to the guidelines provided by the professionals of the SNPC, the patrolling configurations should be as efficient as possible and, at the same time, they should distribute the workload homogeneously among the patrol sectors. Unfortunately, there might be a trade-off between these requirements. The objective function of the MC-PDP takes into consideration the preferences of the decision maker for these factors.

$$\min\ obj(P) = \lambda \cdot \max_{s \in P} \{W^s\} + (1-\lambda) \cdot \frac{\sum_{s \in P} W^s}{p}. \tag{4.8}$$

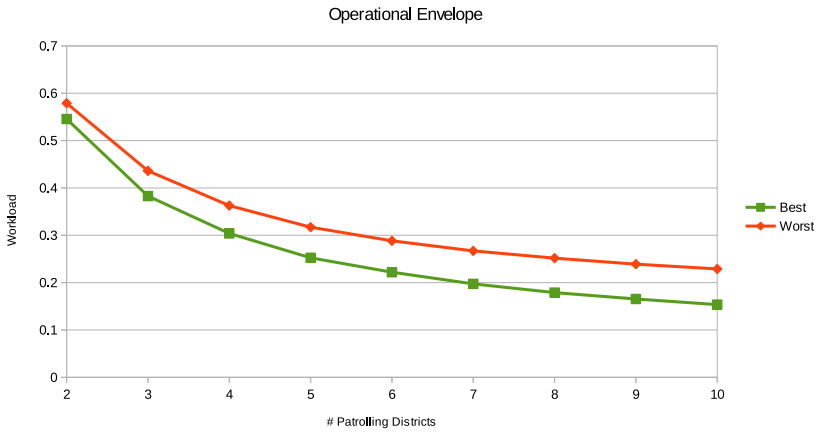
The term $\max_{s \in P}\{W^s\}$ represents the worst workload, while the term $\frac{\sum_{s \in P} W^s}{p}$ is the average workload. This objective function allows the decision maker to examine the trade-off between optimization and balance by a parametric analysis. In fact, by varying λ, the model gives a range from optimization ($\lambda = 0$) to balance ($\lambda = 1$).

Solving the MC-PDP

The MC-PDP is an extremely complex model that cannot easily be solved to optimality. In fact, not only does modeling the property of subset connectivity make the MC-PDP intractable in large problems, but also no linear formulation for the convexity condition has been presented in the literature. Given that computational time is critical — the user expects a solution within a reasonable time — the MC-PDP is solved by a Greedy Randomized Adaptive Search Procedure (GRASP) algorithm. This methodology, thoroughly described in Chapter 3 and

tested in Chapter 6, is capable of generating more efficient patrolling configurations than those are currently being adopted by the SNPC, within one minute.

Figure 4.2. Operational envelope. Workload of the system as a function of the number of patrolling sectors p.



4.2. The Operational Envelope

The MC-PDP finds a good partition of the territory, according to different attributes. When considering suboptimal ways of subdividing a district, the basic question is what is the loss of efficiency involved in terms of workload. We can approximate this loss of efficiency by calculating the distance from the best solution value found. For a fixed number of patrolling sectors and weights, we can represent the increase in workload (or loss of system efficiency) as shown in Figure 4.2.

The operational envelope presented refers to Saturday, 10/13/2012, night shift in the Central District of Madrid (see Section 6.2) and was computed using the real crime distribution in the district. However, we can obtain an approximate operational envelope by using the forecast crime distribution provided by the CRFU. In Figure 4.2, the values on the x-axis represent the number of patrolling sectors. The values on the y-axis display the workload, computed as in Equation (5.8). For this illustrative example, we assign to the weights and the balance coefficient the following values, $(w_\alpha, w_\beta, w_\gamma, w_\delta) = (0.45, 0.05, 0.45, 0.05)$ and $\lambda = 0.1$. As the number of patrolling sectors increases, the workload is consequently decreased. Figure 4.2 has two trends: the upper trend displays the worst

case workload (i.e., the workload obtained when the least effective patrolling plan is implemented) whereas the lower trend depicts the best case workload (i.e., the workload when implementing the best patrolling plan for different values of p). It can be easily seen that the two trends define a range of losses, from the best case to the worst case, that encompasses all the possible ways of protecting the district. This region is referred to as the operational envelope. Knowing the structure of the envelope can be helpful for patrol planning decisions. The lower curve represents a situation of complete control. Thus, the optimal patrolling strategy can be devised. On the other hand, the upper curve shows the effects of applying the worst possible patrolling scheme. The thickness of the envelope provides valuable information regarding the range of the impact of different partitioning strategies using the same amount of resources, and the extent to which the workload may be unnecessarily increased if suboptimal plans are implemented. Prior examples of depicting similar envelopes in other application settings can be found in Urban and Keitt [104], Kim and O'Kelly [56], and Church and Scaparra [26].

4.2.1. Computing the Operational Envelope

An algorithm for the computation of the operational envelope is presented in Algorithm 4.

Computing (or approximating) the operational envelope for a turn is computationally expensive. In fact, the lower trend points are the best solutions found by the MC-PDP, while the upper trend points are the greatest solutions found by a maximization version of the MC-PDP. For the determination of the operational envelope in Figure 4.2, we ran the optimization algorithms for each value of p iteratively and stopped the execution when there was no improvement in the best solution for 10 consecutive iterations. This stopping criterion is often applied in exploration-based optimization algorithms such as Genetic Algorithms to ensure that the method has converged [54, 88]. In terms of computational time, it took approximately two hours to compute the operational envelope presented in Figure 4.2, which is quite time consuming. However, since the operational envelope is a strategical tool that needs to be calculated only once per shift, this computational time is reasonable.

Algorithm 4 Algorithm for the computation of the operational envelope.

Require: R, A, $\mathbf{w}$, λ
for $p \in \{2, \ldots, 10\}$ **do**
  ▷ Obtaining lower trend point
  $i \leftarrow 0$
  $obj\left(P_p^+\right) \leftarrow \inf$
  while $i < 10$ **do**
    solve $obj\,(P) \leftarrow \min$ MC-PDP$(R, A, \mathbf{w}, p, \lambda)$
    if $obj\,(P) < obj\left(P_p^+\right)$ **then**
      $obj\left(P_p^+\right) \leftarrow obj\,(P)$
      $i \leftarrow 0$
    else
      $i \leftarrow i + 1$
    end if
  end while
  ▷ Obtaining upper trend point
  $i \leftarrow 0$
  $obj\left(P_p^-\right) \leftarrow -\inf$
  while $i < 10$ **do**
    solve $obj\,(P) \leftarrow \max$ MC-PDP$(R, A, \mathbf{w}, p, \lambda)$
    if $obj\,(P) > obj\left(P_p^-\right)$ **then**
      $obj\left(P_p^-\right) \leftarrow obj\,(P)$
      $i \leftarrow 0$
    else
      $i \leftarrow i + 1$
    end if
  end while
end for
return $obj\left(P_p^+\right)$ and $obj\left(P_p^-\right)$, $\forall p = 2, \ldots, 10$

4.2.2. Calculating the Efficiency Loss

The decision-maker can exploit the information provided by the operational envelope to compare alternative patrolling configurations and evaluate the potential benefits of corrective plans, such as changes in the number of patrolling sectors, or investments in the analysis of the system to acquire a better definition of the data and the parameters. In fact, the operational envelope can be used to compute the percentage efficiency loss associated to a certain partition P:

$$\text{Efficiency loss} = 100\frac{obj\,(P) - obj\,(P^{+})}{obj\,(P^{-}) - obj\,(P^{+})}, \tag{4.9}$$

where $P^{+}$and is the best and P^{-} the worst known partition for the set of attributes.

CHAPTER 5

LOCAL SEARCH METHODS FOR THE MULTI-CRITERIA POLICE DISTRICTING PROBLEM ON GRAPH

The Multi-Criteria Police Districting Problem (MC-PDP) concerns the definition of sound patrolling sectors in a police district. The model was originally formulated in collaboration with the Spanish National Police Corps and was solved by means of a Steepest Descent Hill Climbing algorithm. One of the major limitations of the MC-PDP is that it requires the territory to be organized as a grid. In this chapter we formulate the MC-PDP for a generic graph, which results in a more applicable and usable model. Also, we propose for its solution three local search algorithms, including a Tabu Search.

For further information on this chapter see [64].

5.1. The Multi-Criteria Police Districting Problem on Graph

The MC-PDP concerns the design of patrol sector configurations that are efficient and that distribute the workload homogeneously among the police officers. A solution to the MC-PDP defined on a graph $G = (N, E)$ is a partition P of the set of nodes N. Each block $A \in P$ of the partition is a connected subset of the node set and represents a patrol sector. Therefore, from this point onward the terms "partition block," "patrol sector" and "sector" will be used interchangeably. The MC-PDP requires the partition blocks to be convex. This condition has been introduced to ensure that all the patrol sector would be intrinsically efficient, i.e., the agent can move within the sector always following the shortest path. Finally, the number of subsets in the partition must be exactly p. The formal elements of the model are presented in the following.

5.1.1. Data and Properties

We define the MC-PDP on a generic graph $G = (N, E)$, with N being the set of nodes and E the set of edges. For each node $i \in N$ the following data is required:

- $a_i \in \mathbb{R}_{\geq 0}$: Total length of the streets to be patrolled at node $i \in N$.
- $r_i \in \mathbb{R}_{\geq 0}$: Risk of crime at node $i \in N$.

Also, each edge $(i, j) \in E$ is characterized by the following:

- $l_{ij} \in \mathbb{R}_{\geq 0}$: Length of edge $(i, j) \in E$.

Finally, $p \in \mathbb{N}_{\geq 2}$ is the number of patrolling sectors to be defined.

Additionally, on the set of nodes N and all of its subsets $N' \subseteq N$ we define the following operations:

- $d_{i,j}(N')$: Shortest path distance between nodes $i, j \in N'$ computed using only the nodes in N'. This distance is calculated considering the length of the edges in the path.
- $d^1_{i,j}(N')$: Shortest edge distance between nodes $i, j \in N'$ computed using only the nodes in N'. This distance is calculated considering exclusively the number of edges in the path.

Given a node subset $N' \subseteq N$, the shortest distances between all the nodes are obtained using the Floyd-Warshall algorithm [34, 107]. The algorithm is initialized with l_{ij} for $d_{i,j}(N')$, and with the adjacency matrix for $d^1_{i,j}(N')$. Other relevant properties defined on the set of nodes N and all of its subsets $N' \subseteq N$ are:

- $\oslash_{N'}$: Diameter of subset N'. The diameter is the maximum distance between two nodes belonging to N', i.e., $\oslash_{N'} = \max_{i,j \in N'} d_{i,j}(N')$.
- $c_{N'}$: Center of subset N'. We define the center of a subset of nodes $N' \subseteq N$ as the node belonging to the subset that minimizes the maximum risk-weighted distance to all the other nodes in the subset. In case of ties, the node that minimizes the sum of the risk-weighted distances is chosen. In summary, $c_{N'} = \arg \text{Lex} \min_{i \in N'} \left(\max_{j \in N'} r_j d_{i,j}(N'), \sum_{j \in N'} r_j d_{i,j}(N') \right)$, where Lex stands for Lexicographic optimization (i.e., hierarchical optimization) of the two objectives. We consider risk-weighted distances as we assume that the agents should spend more time patrolling the nodes having greater risk.

5.1.2. Patrol Sector Attributes and Workload

The MC-PDP evaluates the patrol sectors defined by a partition according to four main attributes: area, isolation, demand, and diameter. All the attributes, explained in the following, are expressed as dimensionless ratios, so as to be comparable.

- **Area**, α^A: This attribute is a measure of the size of the territory that an agent should patrol. It is expressed as the ratio of the area encompassed by patrol sector A to the whole district area.

$$\alpha^A = \frac{\sum_{i \in A} a_i}{\sum_{i \in N} a_i}, \tag{5.1}$$

- **Isolation**, β^A: In the MC-PDP, two patrol sectors support each other if the distance between their centers is less than or equal to a defined constant, K. The value of K can be provided by an expert. Alternatively, for the MC-PDP on graph we recommend the following:

$$K = \frac{\oslash_N}{2\sqrt{p}}, \tag{5.2}$$

that is, we suggest K to be set equal to the total diameter of the graph divided by twice the square root of the number of subsets to be defined. The support received by a patrol sector can be calculated by:

$$b^A = \left| \{ B \in P \, | d_{c_A c_B}(N) \leq K, \ A \neq B \} \right|, \tag{5.3}$$

that is, the support b^A is equal to the number of sectors whose centers are within a distance of K from the center of the currently considered subset. Therefore, the isolation of sector A is computed as:

$$\beta^A = \frac{p - 1 - b^A}{p - 1}, \tag{5.4}$$

- **Risk**, γ^A: This attribute is a measure of the total risk associated to the sector that an agent patrols. It is expressed as the ratio of the total risk of

sector A to the whole district risk.

$$\gamma^A = \frac{\sum_{i \in A} r_i}{\sum_{i \in N} r_i}, \tag{5.5}$$

- **Diameter**, δ^A: The diameter has been introduced in the MC-PDP as an efficiency measure. In fact, the diameter can be interpreted as the maximum distance that the agent associated to the sector would have to travel in case of an emergency call. Therefore, a small diameter results in a low response time. The diameter measure used to evaluate a patrol sector is the ratio of the subset diameter to the diameter of the graph, that is, the maximum possible diameter.

$$\delta^A = \frac{\oslash_A}{\oslash_N}, \tag{5.6}$$

The decision maker can express her preference on each attribute by defining a normalized vector of weights $\mathbf{w} \in \mathbb{R}^4$. By linearly combining the attributes with the preference weights $\mathbf{w}$ we can compute a measure of the workload W^A of a sector A as

$$W^A = w_\alpha \cdot \alpha^A + w_\beta \cdot \beta^A + w_\gamma \cdot \gamma^A + w_\delta \cdot \delta^A. \tag{5.7}$$

5.1.3. Objective Function

The objective of the MC-PDP is to generate patrolling configurations that are efficient and, at the same time, that distribute the workload homogeneously among the patrol sectors. The objective function of the MC-PDP takes into consideration the preferences of the decision maker for these factors by introducing the coefficient $\lambda \in \mathbb{R}$, $0 \leq \lambda \leq 1$, that expresses the decision maker's preference between optimization and workload balance.

$$obj(P) = \lambda \cdot \max_{A \in P} \left\{ W^A \right\} + (1 - \lambda) \cdot \frac{\sum_{A \in P} W^A}{p}. \tag{5.8}$$

The term $\max_{A \in P}\{W^A\}$ represents the worst workload, while the term $\frac{\sum_{A \in P} W^A}{p}$ is the average workload. This objective function allows the decision maker to examine the trade-off between optimization and balance by a parametric analysis. In fact, by varying λ, the model gives a range from optimization ($\lambda = 0$) to balance ($\lambda = 1$).

5.1.4. Problem Formulation

We can now present a mathematical formulation for the MC-PDP on graphs:

$$\begin{aligned}
\min \quad & obj\,(P) && (5.9)\\
\text{s.t.} \quad & \emptyset \notin P && (5.10)\\
& \textstyle\bigcup_{A\in P} A = N && (5.11)\\
& A \cap B = \emptyset \quad \forall A, B \in P \,|\, A \neq B && (5.12)\\
& |P| = p && (5.13)\\
& Conn\,(A) = 1 \quad \forall A \in P && (5.14)\\
& Conv\,(A) = 1 \quad \forall A \in P && (5.15)
\end{aligned}$$

The model optimizes the objective function (5.8). The constraints (5.10)-(5.12) represent the conditions held by a partition P defined on N: P should not contain the empty set $\emptyset$ (5.10), the partition blocks cover N (5.11) and are pairwise disjoint (5.12). The restriction (5.13) concerns the partition cardinality and enforces the number of partition blocks to be exactly p. Conditions (5.14) and (5.15) regard the geometry of the patrol sectors. In fact, $Conn\,(A)$ is an indicator function that equals 1 when A is connected and zero otherwise, and $Conv\,(A)$ is an indicator function that equals 1 when A is convex and zero otherwise. The model establishes that only connected partition blocks are feasible. This condition implies that an agent cannot be assigned to a patrol sector composed of two or more separate areas of the city. Furthermore, all the partition blocks are required to be convex. When a subset is convex, it is also optimally efficient in terms of distance between the points. In fact, in a convex subset there is a minimal shortest path connecting any pair of points. Therefore, this condition allows for the generation of patrol sectors that are more efficient in terms of movement inside of the area. In the following, we illustrate more in detail the concept of graph convexity.

5.1.5. A Note on Graph Convexity and on Convex Graph Partitioning

Let $G = (N, E)$ be a finite simple graph. Let $A \subseteq N$, its closed interval $I[A]$ is the set of all nodes lying on shortest paths between any pair of nodes of A. The set A is convex if $I[A] = A$. In this work, the following equivalent condition is applied:

$$d^1_{i,j}(A) = d^1_{i,j}(N)\ \forall i, j \in A \iff Conv(A) = 1. \tag{5.16}$$

Lemma. *Equation (5.16) is a proper condition for set convexity.*

Proof. Let A be a non convex set. It follows from the definition that $I[A] \neq A$. Let us consider nodes $i, j \in A$ and a node $k \in N$ such that $k \in I[A]$ and $k \notin A$. It follows that $d^1_{i,j}(A) > d^1_{i,j}(N)$. In fact, if it were that $d^1_{i,j}(A) = d^1_{i,j}(N)$ then k would need to belong to A. Now let A be a convex set. It follows from the definition that $I[A] = A$. More specifically, all the nodes lying on the shortest path between between $i, j \in A$ also belong to A. It follows that, necessarily, $d^1_{i,j}(A) = d^1_{i,j}(N)$. □

As we do not make any assumption on the graph G, convexity for all the patrol sectors could not always be possible. In order to always obtain a solution, we relax constraint (5.15) and penalize its violation in the objective function by means of a Lagrange multiplier. The resulting program is:

$$\min \quad \overline{obj}(P) = obj(P) + \mu \sum_{A \in P} (1 - Conv(A)) \tag{5.17}$$

$$\text{s.t.} \quad (5.10) - (5.14) \tag{5.18}$$

The coefficient μ is the Lagrange multiplier associated to the convexity constraint (5.15). We suggest setting $\mu > 1$. In fact, as $obj(P) \leq 1$, setting $\mu > 1$ translates into always preferring a convex graph partition over a non-convex one, regardless of the value of $obj(P)$.

Algorithm 5 Local Search algorithm pseudocode.

```
procedure LocalSearch(P_0)
P* ← P_0;                          ▷ Initialize the best solution found to the initial solution.
t ← 0;
while ¬TerminationCriteria() do
    P_{t+1} ← SelectNeighbor(P_t);          ▷ Select a neighboring solution.
    if P_{t+1} better than P* then
        P* ← P_{t+1};                       ▷ Save the best solution found so far.
    end if
    t ← t + 1;                              ▷ Increase the iteration counter.
end while
return P*;
end procedure
```

5.2. Local Search Methods for the MC-PDP

Local search algorithms move from solution to solution in the space of candidate solutions (the search space) by applying local changes, until certain termination criteria are satisfied, e.g., a solution deemed optimal is found or a time bound is elapsed. One of the main advantages of local search algorithms is that they are *anytime algorithms*, which means that they can return a valid solution even if they are interrupted at any time before they end. For this reason, they are often used to tackle hard optimization problems in a real-time environment, such as the MC-PDP. A generic pseudocode for a local search algorithm is presented in Algorithm 5.

The procedure starts the search from a given initial solution P_0 and it iteratively moves to a solution belonging to the neighborhood of the incumbent one, until a certain termination criteria is met. Different implementations of $TerminationCriteria()$ and $SelectNeighbor()$ result in different local search algorithms. The characteristics of the algorithms developed in this research are presented in the following.

Simple Hill Climbing

At each iteration, the Simple Hill Climbing (SHC) algorithm [87] explores the neighborhood of the incumbent solution to find a better one. The neighborhood of a solution is the set of solutions that can be obtained from the current one by changing it slightly. In this research, we consider all the solutions that can be

obtained by removing a node from a patrol sector and assigning it to another one, without violating constraints (5.10)-(5.14). In our SHC, the $SelectNeighbor\,(P_t)$ procedure explores the neighborhood of the partition P_t in a random fashion and returns the first improving solution found. The algorithm terminates when no improving solution is found or the time limit is exceeded.

Steepest Descent Hill Climbing

The Steepest Descent Hill Climbing (SDHC) algorithm [87] is a variant of the SHC that explores the whole neighborhood of the incumbent solution and chooses the best solution belonging to it. This is the same algorithm originally proposed for the solution of the MC-PDP [17].

Tabu Search

Similarly to the SDHC, the Tabu Search (TS) algorithm [39, 40] explores the whole neighborhood of the incumbent solution. However, the TS chooses for the next iteration the best solution found that is not *tabu*. Also, the TS does not terminate if an improving solution is not found. This allows the algorithm to escape local optima. The criterion that is used to declare a certain point of the neighborhood as tabu is based on a short-term memory. At each iteration, the TS presented in this thesis stores the current solution in the short-term memory with an associated expiration counter initially set to T. During the exploration of a neighborhood all the solutions found that are already included in the short term memory are marked as tabu and their expiration counter are reset to T. Finally, at the end of the iteration, all the expiration counters are decreased by one and the solutions whose counters have reached zero are removed from the short term memory.

The algorithm terminates when the time limit is exceeded, when no non-tabu solution is found in the current neighborhood, or after a fixed number I of non-improving iterations. We suggest setting the parameters T and I to the cardinality of the node set, i.e., $T = I = |N|$.

5.2.1. Multi-Start Local Search Algorithms

Local search methods are very good at exploring certain zones of the solution space but they generally end up in local optima. Multi-start is a very simple and general diversification method. In order to better explore distant portions of the solution space the search is started more than one time from different points. The pseudocode of a multi-start procedure is illustrated in Algorithm 6.

Algorithm 6 Multi-start pseudocode.

```
procedure MultiStart()
while ¬TerminationCriteria() do
    P ← InitialSolution();          ▷ Generate an initial solution.
    P' ← LocalSearch(P);            ▷ Improve the current solution.
    if P' better than P* then
        P* ← P';                    ▷ Save the best solution found so far.
    end if
end while
return P*;
end procedure
```

The procedure alternates a solution generation procedure with a local search step, until the termination criteria, e.g. a time limit is exceeded.

Generating an Initial Solution

To generate an initial solution at each iteration of the multi-start algorithm, we use the random greedy algorithm proposed in Chapter 3, adapted to work on a generic graph. In summary, the algorithm generates a solution by randomly choosing the first node of each sector and then expanding the sectors in a greedy fashion while preserving their connectivity. Initially, the partition blocks are empty. In the first phase of the algorithm, each block is initialized with a randomly chosen node. Subsequently, at each iteration of the second phase, the algorithm extends the initial solution by assigning a node to a single sector. The algorithm chooses the combination of node and sector that results in the best feasible solution. The algorithm ends when all the points have been assigned to subsets. It is important to notice that, in the current version of the algorithm, the solutions are evaluated by using the relaxed objective function (5.17) .

CHAPTER 6

COMPUTATIONAL EXPERIMENTS AND RESULTS.

6.1. Results for MCPDP Applied to a Case Study of the Central District of Madrid

The Local Search Algorithm has been applied and tested on a case study of the Central District of Madrid. The solutions identified by the optimization algorithm have been analyzed and compared to the standard patrolling configurations currently adopted by inspectors of the SNPC.

6.1.1. The Central District of Madrid

Madrid is the capital of Spain and the most populous city in the country with 3,207,247 inhabitants as of 2013. In the metropolitan area as a whole, the population is 6,543,031. The Central District of Madrid, on which we focus our work, has an area of more than two square miles and comprises six neighborhoods: Palacio, Embajadores, Cortes, Justicia, Universidad, and Sol. Its population is approximately 150,000 inhabitants.

Datasets

To determine the best grid size, we can take advantage of the results of Gorr and Harries [41]. In fact, the authors show that the average monthly crime counts for each cell of the grid needs to be on the order of 30 or more to achieve good forecast accuracy. The resulting grid for the Central District of Madrid has nine rows and nine columns, and can be seen in Figure 6.1. Crime analysts from the SNPC stated that the grid is sufficiently precise for the determination of patrol districts.

In this case study, we consider the thefts committed during the month of October, 2011. Theft is the most frequent type of crime committed in Spain and one of the main priorities for the SNPC is its reduction. The month of October has been chosen, as it is an "average" month in terms of population and activity and it has only one holiday. More specifically, we consider the following working shifts:

- SATT3: Saturday, 10/15/2011, night shift (10 PM–8 AM).
- SUNT1: Sunday, 10/16/2011, morning shift (8 AM–3 PM).
- MONT2: Monday, 10/17/2011, afternoon shift (3 PM–10 PM).

These three shifts have been chosen for their representativeness, as crime activity varies by time of day, day of the week, and by sector, and exhibits seasonal effects [28]. Figure 6.1 illustrates the distribution of thefts in the three shifts considered. SATT3 is characterized by a high level of nightlife, with people coming from other districts of Madrid as well as other cities. In the picture it can be seen that thefts are committed in almost all the territory, with the highest levels concentrated around Plaza Mayor, the central plaza of the city. SUNT1 has a moderate level of criminality, mostly concentrated in the south of the district where a very popular flea market (El Rastro) is held every Sunday morning. Finally, MONT2 presents the characteristics of a normal business day, with low levels of criminal activity, mostly concentrated in the commercial area.

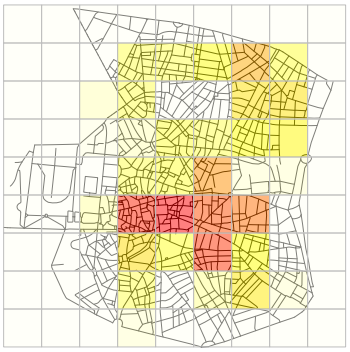
6.1.2. Current Patrolling Configurations Analysis

During an interview, a service coordinator in charge of the patrolling operations of the Central District of Madrid stated that, on a "normal day", one of the following patrol sector configurations is applied:

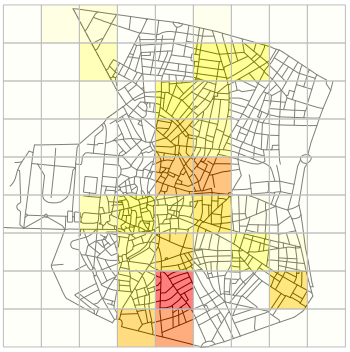
- CONF2: The district is divided into two big sectors by the Gran Via, the main artery in the territory, and the agents are free to patrol the assigned area *ad libitum*. The northern sector includes two neighborhoods (i.e., Universidad and Justicia) while the southern sector includes four neighborhoods (i.e., Palacio, Sol, Embajadores, and Cortes).
- CONF6: The district is partitioned according to its six neighborhoods.

Figure 6.1. Number of thefts in the Central District of Madrid. Red represents a high crime level, while white represents no criminal activity.

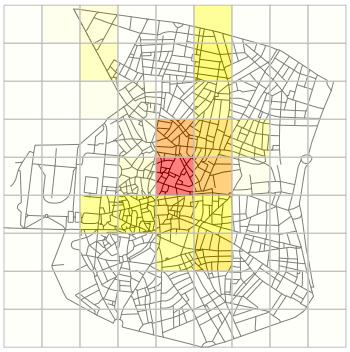
(a) SATT3: Saturday, 10/15/2011, night shift (10 PM–8 AM).



(b) SUNT1: Sunday, 10/16/2011, morning shift (8 AM–3 PM).



(c) MONT2: Monday, 10/17/2011, afternoon shift (3 PM–10 PM).



To be able to compare the performance of these configurations with those identified by the optimization algorithm, we represented CONF6 and CONF2 using the same grid structure adopted by the optimization algorithm, as illustrated in Figures 6.2 and 6.3. The cells of the grid that are shared by more than one sector have been assigned to the sector occupying the most of its area. It is important to highlight that in both configurations there exists one sector that is not convex, i.e., the green sector in CONF6 and the light blue sector in CONF2. Therefore, the configuration currently adopted by the SNPC would be infeasible according to the optimization model proposed. This might result in better attribute values for these solutions than those achievable with a feasible solution. In the following, we use these configurations as a comparative basis for the quality of the solutions identified by the optimization algorithm.

6.1.3. Analysis of the Optimization Model Solutions

We now analyze the quality of the solutions found by the optimization algorithm, by comparing them to the patrolling configurations currently adopted by the SNPC. The optimization algorithm was implemented in C++. The experiments were run on a computer with an Intel Core i5-2500K CPU having four cores at 3.30GHz and 4GB RAM . The program was run on only one core and the measured RAM memory use is less than 2MB. Given that the police district optimizer should be part of a DSS and, therefore, be sufficiently interactive, the computational time limit for each test was set to 60 seconds. Concerning the parameters, we asked a service coordinator in charge of the patrolling operations of the Central District to define her preferences among the criteria and the values for the weights and the parameter λ. The parameter values adopted in the experiments are the following:

- Dataset: {SATT3, SUNT1, MONT2}.
- Number of patrol sectors, p: $\{2, 6\}$.
- Preference weights, $(w_\alpha, w_\beta, w_\gamma, w_\delta)$: $\{(0.45, 0.05, 0.45, 0.05)\}$.
- Balance coefficient, λ: $\{0.1\}$.

In any event, the algorithm can be run for any feasible combination of the parameters. In the following, we compare the solutions found by the proposed algorithm and the patrolling configurations currently adopted by the SNPC.

Table 6.1. Comparison of the patrolling configurations currently adopted by the SNPC with those generated by the optimization algorithm. The tables show the solution values, the attribute values, and the percentage improvement of the algorithm solutions over the current patrolling configuration.

(a) Scenario with two patrol sectors.

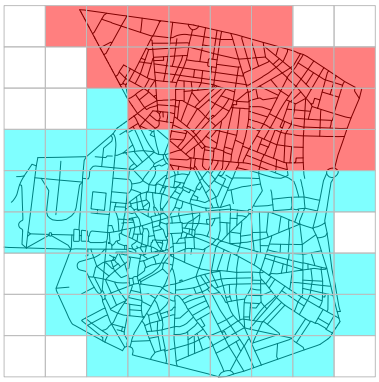
Dataset	Method	$Obj\ (\overline{P})$	Max Area	Avg Support	Min Support	Max Demand	Avg Diameter	Max Diameter
SATT3	CONF2	0.56	83588	0	0	33.4	11	12
	Algorithm 3	[0.49, 0.49]	[66247, 66247]	[1, 1]	[1, 1]	[24.59, 24.59]	[11.5, 11.5]	[12, 12]
	Improvement	[12.5%, 12.5%]	[20.75%, 20.75%]	∞	∞	[26.38%, 26.38%]	[-4.55%, -4.55%]	[0, 0]
SUNT1	CONF2	0.56	83588	0	0	19.64	11	12
	Algorithm 3	[0.49, 0.49]	[70728, 70728]	[1, 1]	[1, 1]	[13.76, 13.76]	[11.5, 11.5]	[12, 12]
	Improvement	[12.5%, 12.5%]	[15.38%, 15.38%]	∞	∞	[29.94%, 29.94%]	[-4.55%, -4.55%]	[0, 0]
MONT2	CONF2	0.55	83588	0	0	12.80	11	12
	Algorithm 3	[0.49, 0.49]	[68002, 68002]	[1, 1]	[1, 1]	[11.42, 11.42]	[11.5, 11.5]	[12, 12]
	Improvement	[10.91%, 10.91%]	[18.65%, 18.65%]	∞	∞	[10.78%, 10.78%]	[-4.55%, -4.55%]	[0, 0]

(b) Scenario with six patrol sectors.

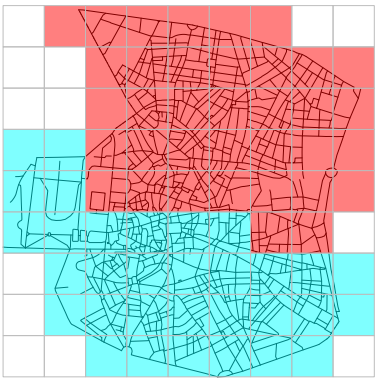
Dataset	Method	$Obj\ (\overline{P})$	Max Area	Avg Support	Min Support	Max Demand	Avg Diameter	Max Diameter
SATT3	CONF6	0.20	33065	2.33	1	8.85	5.17	7
	Algorithm 3	[0.19, 0.19]	[28281.22, 30050.94]	[3.66, 3.75]	[2.65, 2.87]	[13.49, 14.06]	[5.43, 5.62]	[8.74, 9.42]
	Improvement	[5%, 5%]	[9.12%, 14.47%]	[57.08%, 60.94%]	[165%, 187%]	[-58.87, -52.43]	[-8.70%, -5.03%]	[-24.86%, -34.57%]
SUNT1	CONF6	0.21	33065	2.33	1	11.21	5.17	7
	Algorithm 3	[0.18, 0.19]	[27851, 29331.08]	[3.64, 3.69]	[2.88, 3]	[6.44, 6.88]	[5.25, 5.35]	[7.29, 7.79]
	Improvement	[9.52%, 14.29%]	[11.29%, 15.77%]	[56.22%, 58.37%]	[188%, 200%]	[38.63, 42.55]	[-3.48%, -1.55%]	[-11.29%, -4.14%]
MONT2	CONF6	0.21	33065	2.33	1	7.48	5.17	7
	Algorithm 3	[0.18, 0.18]	[32122.76, 33319.8]	[3.81, 3.89]	[2.68, 2.88]	[6.71, 7.16]	[5.24, 5.36]	[9.15, 9.61]
	Improvement	[14.29%, 14.29%]	[-0.77%, 2.85]	[63.52%, 66.95%]	[168%, 188%]	[10.29%, 4.28%]	[-3.68%, -1.35%]	[-37.29%, -30.71%]

Figure 6.2. Comparison of the patrolling configurations currently adopted by the SNPC with those generated by the optimization algorithm. Scenario with two patrol sectors. Each sector is represented in a different color.

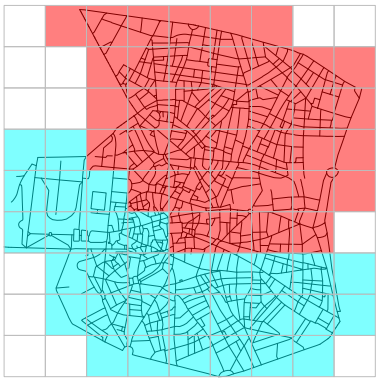
(a) CONF2. Solution currently adopted by the SNPC.



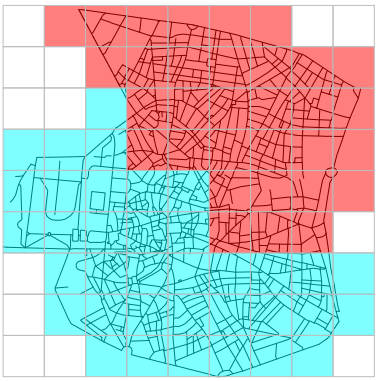
(b) SATT3. Best solution found by the random search algorithm.



(c) SUNT1. Best solution found by the random search algorithm.



(d) MONT2. Best solution found by the random search algorithm.



Scenario with Two Patrol Sectors

As the optimization algorithm is random in nature, we ran each configuration 50 times. The best solutions found by the algorithm are displayed in Figure 6.2. According to the figures, the optimization algorithm assigns a greater area to the northern sector than the current solution of the SNPC. Also, we can see that the northern sector slowly decreases in size as we move from the Saturday night shift to the Monday afternoon shift, to adapt to the changes in the crime activity level and distribution.

The solution and the attribute values are illustrated and compared in Table

6.1a. The first three columns report the dataset, the methodology, and the objective function value. Then, for each attribute, the average and the worst value are given. The area and demand averages are not shown, as they are constant. For the algorithm, we show the 95% confidence interval computed over the 50 runs. Also, to simplify the interpretation of the differences in the attributes values, we show the percentage improvement of our solutions over the current solution adopted by the SNPC. The improvement was calculated as $100 \cdot (1 - \frac{Z_{\text{ALG}}}{Z_{\text{SNPC}}})$, except for the average and min support, that were calculated as $100 \cdot (\frac{Z_{\text{ALG}}}{Z_{\text{SNPC}}} - 1)$, where Z_{ALG} is the value of the solution computed by the optimization algorithm and Z_{SNPC} is the value for the current patrolling configuration in use by the SNPC. In the instances considered, the proposed algorithm produces patrolling configurations that are always better than the current one in terms of the objective function, with an average improvement of 11.97%. Also, we can see that all the attributes experience a significant improvement, with the exception of the diameter, which worsens by 4.55% on average.

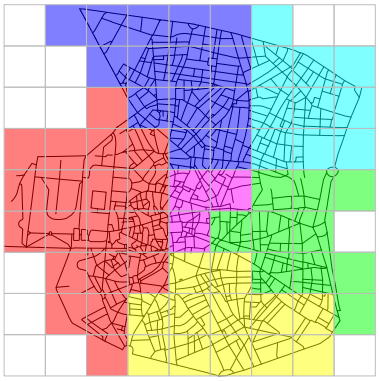
Scenario with Six Patrol Sectors

The best solutions found by the algorithm with six sectors are shown in Figure 6.3. We can see that there are significant differences between them and the current patrolling configuration. From the observation of the configurations with six subsets, one can see the importance of designing patrolling districts tailored for the specific characteristics of each shift. In fact, we can see that the size and location of the sectors changes to adapt to the crime distribution in each shift. For the Saturday night shift (Figure 6.3b), the focus is on the center of the district, where most of the nightlife takes place. On Sunday morning (Figure 6.3c), as expected, we can see that most of the agents should be located on the southern part of the district, where the flea market takes place. On the other hand, on Monday afternoon (Figure 6.3d), patrolling in the southern part of the district can be reduced (only two sectors), in favor of a greater control of the central and the northern parts of the district, where the commercial activities are located.

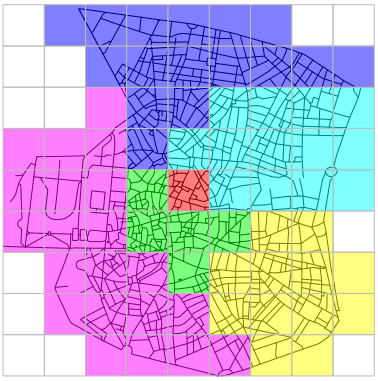
The solution and the attribute values are illustrated and compared in Table 6.1b. Also, in the scenario with six patrol sectors the algorithm generates better partitions than those currently in use in the SNPC, with an average improvement of 10.40%. In fact, it can be seen that the objective function value of the current

Figure 6.3. Comparison of the patrolling configurations currently adopted by the SNPC with those generated by the optimization algorithm. Scenario with six patrol sectors. Each sector is represented in a different color.

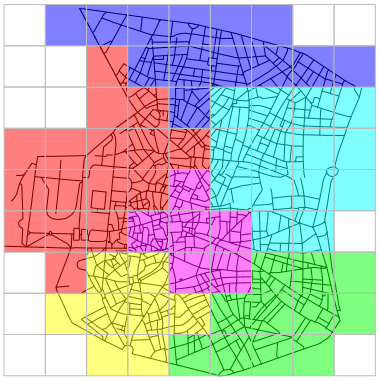
(a) CONF6. Solution currently adopted by the SNPC.



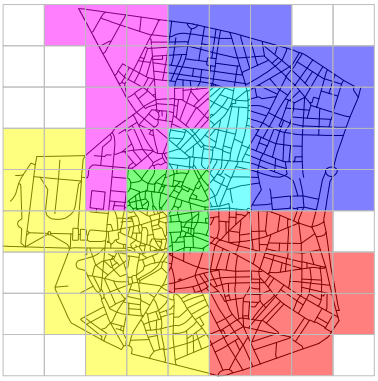
(b) SATT3. Best solution found by the random search algorithm.



(c) SUNT1. Best solution found by the random search algorithm.



(d) MONT2. Best solution found by the random search algorithm.



configuration is always larger that of the configurations generated by the optimization algorithm. Also, the optimization algorithm improves notably the average and minimum support, while keeping the max area and the max demand below the values of the current solutions. The only exception is for dataset SATT3, where the max demand is much higher than that of the current solution.

6.2. A DSS for Predictive Police Patrolling Tested with a Case Study in the Central District of Madrid

To test the P^3-DSS, we developed an initial version considering the thefts reported during the years 2008 to 2012 in the Central District of Madrid. The dataset includes exactly 105,755 incidents. This case study focuses on theft as it is the single most frequent type of crime committed in Spain and one of the main priorities for the SNPC is its reduction. However, extending the P^3-DSS to other districts and to consider several types of crime is straightforward and can be accomplished with little change in the structure of the units and the models. Nevertheless, the final version of the P^3-DSS will require dedicated hardware to be able to cover the whole national territory.

6.2.1. Overview of the Central District of Madrid

The population of the Central District of Madrid is extremely heterogeneous; there is also a large transient population that increasingly commutes to this district for reasons of work, sightseeing, or leisure.

In Spain, the security of towns is the responsibility of the SNPC, usually sharing the territory with other local security forces. Currently, the inspector in charge of civil protection operations in a shift decides the distribution of agents in the district. This decision is normally taken considering mostly their personal experience, their intuition, and also some descriptive statistics, such as the summary of the criminal activity of the last days.

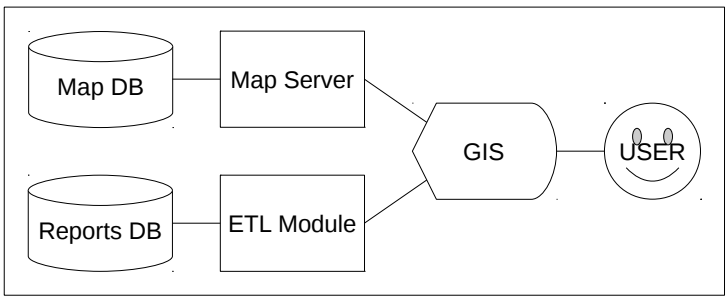
6.2.2. Implementation and Integration of P^3-DSS

The DPPU and the CRFU have been developed in R [81] and then embedded in C++. For their implementation, the following R packages have been

used: sp [6, 79], rgeos [5], maptools [4], and forecast [49]. The PSOU has been programmed entirely in C++.

The GIS currently in use by the SNPC (SNPC-GIS) is structured as shown in Figure 6.4.

Figure 6.4. SNPC-GIS diagram. A line connecting two elements indicates bi-directional communication between them.



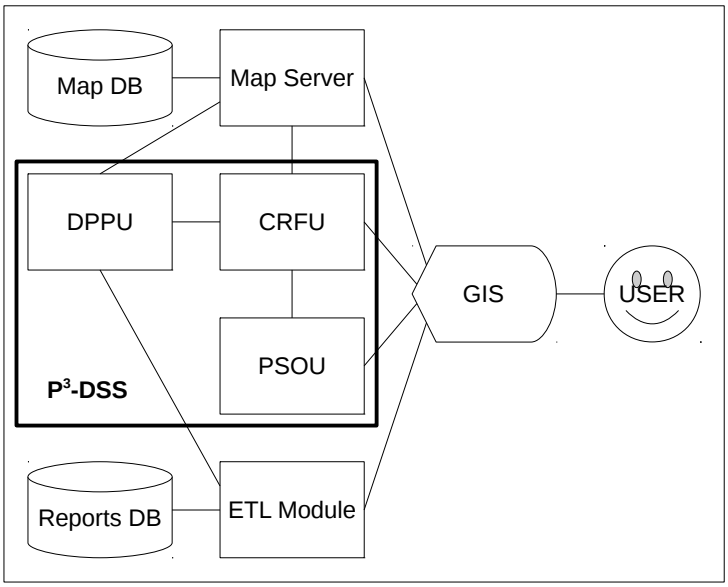
The SNPC-GIS is composed of the following elements:

- Map Database and Server: These units provide access to the updated road maps of Spain.
- Reports Database and ETL Module: These units provide access to crime records.
- GIS: The GIS in use in the SNPC allows visualizing crime records on the map and representing the location of police vehicles in the territory.

The P^3-DSS proposed in this thesis is composed of three fundamental parts that interact naturally with the existing SNPC-IS, as illustrated in Figure 6.5.

The DPPU interacts with the ETL Module to access the Reports DB, get the crime records data, and to combine this data with the geographical information returned by the Map Server to build the crime risk matrix C. This data is passed to the CRFU to forecast crime risk levels for future shifts. Both the DPPU and the CRFU need to regularly update their data structure and models so as to always have the best forecasting quality. The PSOU obtains the forecast crime risk levels from the CRFU, and the user's preferences from the GIS. Finally, the GIS is connected to the CRFU and the PSOU to make queries regarding the distribution of crime risk in a future shift and the recommended patrol sector configuration, respectively. The GIS has been updated to visualize the patrol configurations,

Figure 6.5. P^3-DSS integrated into the SNPC Information System. A line connecting two elements indicates bi-directional communication between them.



while the forecast crime risk levels are represented using an existing feature for the display of heat maps.

6.2.3. Crime Risk Prediction Quality

We now analyze the quality of the crime risk forecast given by the CRFU. The seasonality period-length was chosen according to preliminary experiments on the dataset that showed that the best performance is obtained when using a seasonality period-length of 21 shifts.

Dataset

We assessed the quality of the CRFU forecasts by computing the forecasting Mean Square Error (MSE), considering all the months in 2012 and three areas of interest in the Central District of Madrid. For each validation period, we trained the forecasting model using all the criminal records prior to the period considered (e.g., for the January 2012 validation period, we trained the forecasting models using the historical data from January 2008 to December 2011). Next, we computed the validation MSE for the following areas of interest of the district:

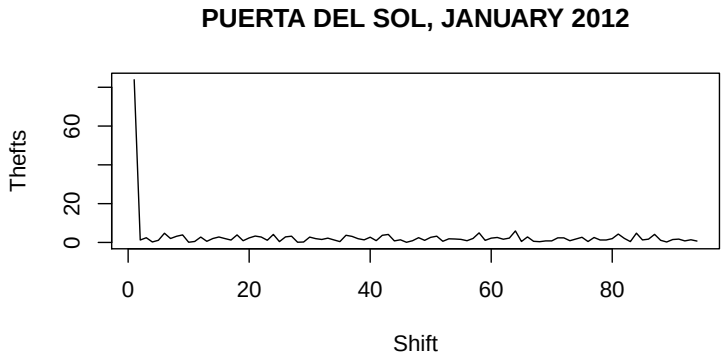
- *Tribunal.* Located in the north of the district, *Tribunal* is a well-frequented crossroad positioned right at the border between two areas that are very popular for nightlife: *Malasaña* and *Chueca.*

- *Puerta del Sol.* One of Madrid's main attractions and a prominent meeting place for citizens and tourists alike, *Puerta del Sol* is a square located in the center of the district.

- *El Rastro.* An area located in the south of the district that hosts a famous flea market every Sunday morning.

Performance Analysis

Figure 6.6. Crime distribution for January 2012.



The training and validation MSEs are given in Table 6.2a. Except for "January 2012/*Puerta del Sol*," the CRFU performed extremely well in validation, with a maximum validation MSE of 2.98, corresponding to an average forecasting error of 1.73 thefts per shift. The "January 2012/*Puerta del Sol*" case can be easily explained by observing Figure 6.6.

In fact, the plot has a peak of more than 80 thefts in the first shift. This observation, that statistically can be considered an outlier, is due to the fact that the first shift of January 2012 corresponds to New Year's Eve. It is traditional in Madrid to celebrate this event in the *Puerta del Sol.* Therefore, the high aggregation of people in the square and the festive atmosphere contributed to the extraordinarily high number of thefts.

Table 6.2. Forecasting MSEs obtained by the CRFU and by the baseline.

(a) Forecasting MSEs obtained by the CRFU.

Area	MSE	01/2012	02/2012	03/2012	04/2012	05/2012	06/2012	07/2012	08/2012	09/2012	10/2012	11/2012	12/2012
Tribunal	Training	0.22168	0.22439	0.23043	0.23004	0.23365	0.23238	0.23144	0.23097	0.23081	0.23104	0.24023	0.24323
	Validation	0.37128	0.53738	0.21372	0.41565	0.18121	0.21446	0.26313	0.23633	0.25698	1.04725	0.48787	0.48897
Puerta del Sol	Training	5.5535	6.91950	6.78993	6.68529	6.60658	6.49690	6.39895	6.30892	6.21466	6.12261	6.04007	5.97210
	Validation	69.19553	1.27499	1.25833	2.69727	1.52356	0.84760	1.56948	1.56276	1.16589	1.89223	1.54950	2.47830
El Rastro	Training	0.10257	0.11009	0.10347	0.10295	0.10307	0.10293	0.10267	0.10296	0.10210	0.10124	0.10040	0.09919
	Validation	0.03804	0.26832	0.07405	0.11660	0.08281	0.08519	0.134499	0.02678	0.04322	0.05627	0.02750	0.11087

(b) Forecasting MSEs obtained using the baseline. The percentages are the ratios of the baseline MSEs to the CRFU MSEs.

Area	MSE	01/2012	02/2012	03/2012	04/2012	05/2012	06/2012	07/2012	08/2012	09/2012	10/2012	11/2012	12/2012
Tribunal	Validation	1.48378	2.07518	1.23134	1.16131	0.51028	0.56323	0.40318	0.40188	0.64585	2.16645	2.04720	1.53111
	Ratio	399.64%	386.17%	576.15%	279.40%	281.60%	262.63%	153.22%	170.05%	251.32%	206.87%	419.62%	235.93%
Puerta del Sol	Validation	134.45202	2.68679	2.74614	4.47852	2.79284	1.71924	3.00656	1.640898	1.71565	3.96208	3.81717	4.73182
	Ratio	194.31%	210.73%	218.24%	166.04%	183.31%	202.84%	191.56%	100.05%	147.15%	209.39%	246.35%	158.88%
El Rastro	Validation	0.18613	1.18519	0.50907	0.94083	0.41429	0.50386	0.56963	0.23759	0.45586	0.64930	0.33392	0.59623
	Ratio	489.30%	441.71%	687.47%	806.89%	500.29%	591.45%	423.52%	887.19%	1054.74%	1153.90%	1214.25%	537.77%

Comparison with the Baseline

To understand the quality of the predictions returned by the CRFU we compare the results obtained against a baseline. Table 6.2b presents the MSEs obtained by a baseline model that predicts tomorrow's crime rate to be the same as today's crime rate. In absolute terms, the baseline model performs fairly well, with relatively low values of validation MSE. We can conclude that, with the exception of the "January 2012/*Puerta del Sol*" case, the crime risk level is quite simple to forecast using only information on time and location, in the dataset considered. In the table, the percentages are the ratios of the MSEs obtained by the baseline model to those of the CRFU. These values confirm that the CRFU provides a much better prediction than the baseline model. In fact, the validation MSE of the baseline model is always larger than the MSE of the CRFU, especially in *El Rastro*, where it is always more than four times larger.

6.2.4. Predictive Police Patrol Configurations Quality

In Section (6.1) we extensively analyzed the performance of the MC-PDP in the presence of perfect information. We now assess the quality of the patrolling configurations generated by the P^3-DSS based on the forecast crime risk.

Comparison with Standard Configurations

To analyze the quality of the solutions found by the optimization algorithm, we compare them to the patrolling configurations currently adopted by the SNPC.

In the Central District of Madrid, on an "average day", one of the following patrol sector configurations is applied:

- CONF6: The district is partitioned according to its six neighborhoods.
- CONF2: The district is split into two big sectors by the *Gran Via*, the main artery in the territory, and the agents are free to patrol the assigned areas ad lib. The northern sector includes two neighborhoods (viz., *Universidad* and *Justicia*) while the southern sector includes four neighborhoods (viz., *Palacio*, *Sol*, *Embajadores*, and *Cortes*).

To be able to compare the performance of these configurations with those determined by the optimization algorithm, we represented CONF6 and CONF2 using the same grid structure adopted by the optimization algorithm, as illustrated in

Figure 6.7. Patrolling configurations currently adopted by the SNPC in the Central District of Madrid. Each sector is represented by a different color.

(a) CONF2: the city is divided into two big sectors by the *Gran Via*, the main street in the district, and the agents are free to patrol the assigned area ad lib.

(b) CONF6: the district is partitioned according to its neighborhoods.

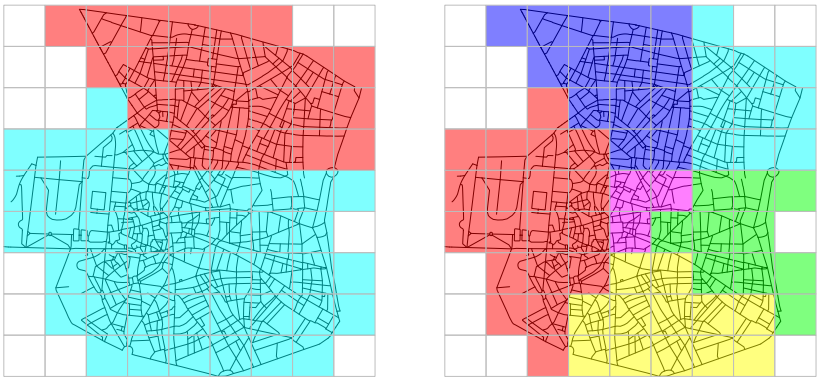


Figure 6.7. The cells of the grid shared by more than one sector have been assigned to the sector occupying most of its area. It should be noticed that both configurations have one sector that is not convex, i.e., the green sector in CONF6 and the light blue sector in CONF2. Therefore, the configurations currently adopted by the SNPC would be infeasible according to the optimization model proposed. This might result in better attribute values for these solutions than those achievable with a feasible solution by the MC-PDP.

To assess the quality of the predictive patrol sector configurations, we ran an analysis composed of the following steps:

1. Generation of 1000 random test instances. Each instance is represented as a tuple $(w_\alpha, w_\beta, w_\gamma, w_\delta, \lambda, \text{shift})$. For each sample, the attribute weights $(w_\alpha, w_\beta, w_\gamma, w_\delta)$ are sampled from a uniform distribution $\mathcal{U}(0,1)$ and normalized, the balance coefficient λ is sampled from a uniform distribution $\mathcal{U}(0,1)$ and the shift is chosen randomly from the set of all the shifts in 2012.

2. Forecasting of the crime risk distribution and extraction of the validation crime distribution for each of the test instances.

3. Generation of patrolling configurations based on the forecast crime risk by running the MC-PDP once, using 2 and 6 patrol sectors ($p = \{2, 6\}$). The

optimization algorithm has been run for 60 seconds to simulate a real-time environment.

4. Evaluation of patrolling configurations generated by the P^3-DSS using the validation data.
5. Evaluation of the standard patrolling configurations currently adopted by the SNPC (CONF2 and CONF6) using the validation data.
6. Statistical comparison of the objective function values obtained on the same test instances by the standard configurations and those generated by the P^3-DSS.

The results of the statistical analysis will be presented in the following. First, we tested the data for each group for normality using a Shapiro–Wilk Normality Test and found that they did not follow a normal distribution. Then, we applied a Friedman Test, a nonparametric test of nonindependent data from two or more groups that does not require the data to proceed from a normal distribution. The statistical difference is significant in both cases. In fact, the p-values are $p = 2.56e - 12$ for the 2 patrol sectors case, and $p = 0.02781$ for the 6 patrol sectors case. We can therefore state that, even in the face of uncertainty, the P^3-DSS produces patrolling configurations that dominate those currently adopted by the SNPC.

Comparing the Loss of Performance

It is difficult to understand what is the real improvement in terms of efficiency resulting from using the P^3-DSS. That is because the objective function in Equation (5.8) includes both a balance term and a global performance term. Fortunately, we can use the operational envelope (see Section 4.2) to compare the patrolling configurations in terms of efficiency loss (Equation 4.9).

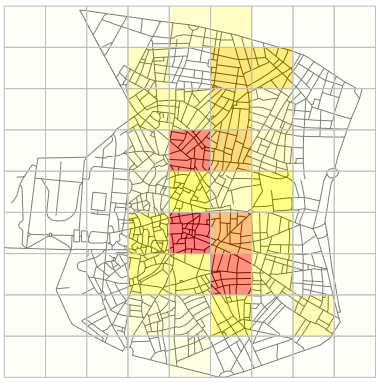
Given the computational time required to compute the operational envelope for each case, this analysis has been limited to three shifts and one configuration of the attributes. The shifts and the attributes have been identified with the help of a service coordinator in charge of the patrolling operations of the Central District of Madrid. In the course of many meetings, we asked the service coordinator to identify a small number of shifts that could be considered as typical scenarios and that presented very different crime activity patterns. The following shifts are

considered:

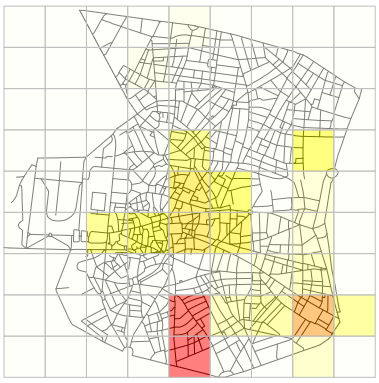
- SATT3: Saturday, 10/13/2012, night shift (10 PM–8 AM).
- SUNT1: Sunday, 10/14/2012, morning shift (8 AM–3 PM).
- MONT2: Monday, 10/15/2012, afternoon shift (3 PM–10 PM).

Figure 6.8. Maps of the number of thefts reported in the Central District of Madrid. The red shade represents a high crime level while the white shade represents no criminal activity.

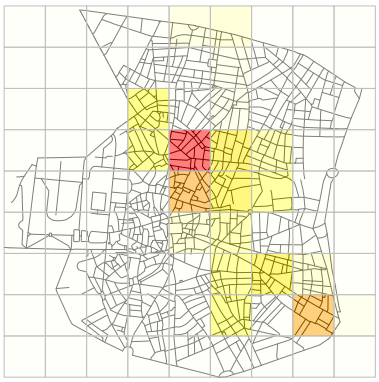
(a) SATT3: Saturday, 10/13/2012, night shift (10 PM–8 AM).



(b) SUNT1: Sunday, 10/14/2012, morning shift (8 AM–3 PM).



(c) MONT2: Monday, 10/15/2012, afternoon shift (3 PM–10 PM).

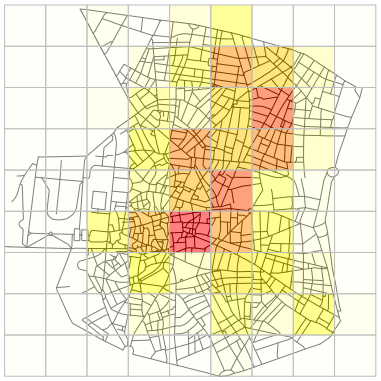


These three shift were chosen for their representativeness. Figure 6.8 illustrates the distribution of thefts in these three shifts. SATT3 is characterized by a high level of nightlife, with people coming from other districts of Madrid as well as other cities. In the picture it can be seen that thefts are committed in almost all the territory, with the highest levels concentrated around *Plaza Callao*, a busy meeting

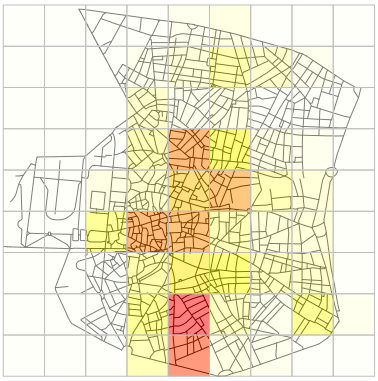
place, *Plaza Mayor*, the central plaza of the city, and *Lavapies*, a difficult area. SUNT1 has a moderate level of criminality, mostly concentrated in the south of the district where the popular *El Rastro* flea market is held every Sunday morning. Finally, MONT2 presents the characteristics of a normal business day, with low levels of criminal activity, mostly concentrated in the commercial area. Figure 6.9

Figure 6.9. Maps of the forecast crime risk in the Central District of Madrid. The red shade represents a high crime risk level while the white shade represents no criminal risk.

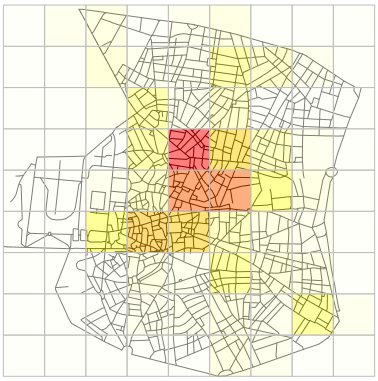
(a) Prediction for SATT3: Saturday, 10/13/2012, night shift (10 PM–8 AM).



(b) Prediction for SUNT1: Sunday, 10/14/2012, morning shift (8 AM–3 PM).



(c) Prediction for MONT2: Monday, 10/15/2012, afternoon shift (3 PM–10 PM).



shows the forecast crime risk distribution obtained by the CRFU. Visually, the real and forecast crime distributions appear to be very close.

Concerning the weights and the balance coefficient, we explained the mean-

ing of each attribute in the model to the service coordinator. Then we asked her to rate from 0 to 10 the importance of each attribute and her preference between patrolling efficiency and workload balance. The following values were obtained after normalizing and scaling the figures provided: • Attribute weights, $(w_\alpha, w_\beta, w_\gamma, w_\delta) = (0.45, 0.05, 0.45, 0.05)$.

• Balance coefficient, $\lambda = 0.1$. The objective function values in Equation (5.8) of the standard patrolling configurations are provided in Table 6.3a.

Table 6.3. Comparison of the objective function values.

(a) Objective function values of the patrol sector configurations currently adopted by the SNPC.

Shift	Configuration	p	$Obj\,(P)$
SATT3	CONF2	2	0.55086
	CONF6	6	0.20655
SUNT1	CONF2	2	0.56515
	CONF6	6	0.21236
MONT2	CONF2	2	0.55074
	CONF6	6	0.20830

(b) Objective function values of the patrol sector configurations obtained by P^3-DSS. 95% confidence intervals over 50 runs.

Dataset	p	$Obj\,(P)$ 95% CI
SATT3	2	[0.54560, 0.54560]
	6	[0.19699, 0.20061]
SUNT1	2	[0.54695, 0.54695]
	6	[0.19703, 0.19973]
MONT2	2	[0.54112, 0.54112]
	6	[0.19399, 0.19708]

The MC-PDP was run for 60 seconds using the parameters provided by the service coordinator. As the optimization algorithm is random in nature, we ran each configuration 50 times, using the forecast data. After that, we evaluated the solutions using the validation data. The 95% confidence intervals of the solution values using the real data are shown in Table 6.3b. As expected, the confidence intervals are better (lower) than the objective function values of the corresponding SNPC configurations in all cases.

To understand the real improvement in terms of efficiency resulting from the adoption of the P^3-DSS, we compute the efficiency loss for both types of patrolling configurations.

Table 6.4 provides the efficiency loss, computed according to Equation (4.9), associated to the patrolling configurations currently adopted by the SNPC and the 95% confidence interval of the efficiency loss relative to the solutions identified by the P^3-DSS. For this experiment the operational envelope was approximated by running the optimization algorithms on the validation data. The results in the

Table 6.4. Comparison of the patrol sector configurations obtained by P^3-DSS with those currently adopted by the SNPC.

Dataset	Configuration	p	SNPC Solutions Efficiency Loss (%)	P^3-DSS Solutions Efficiency Loss (%) 95% CI
SATT3	CONF2	2	15.77	[0,0]
	CONF6	6	20.01	[10.15,14.38]
SUNT1	CONF2	2	82.75	[0,0]
	CONF6	6	24.32	[8.81,12.04]
MONT2	CONF2	2	24.14	[0,0]
	CONF6	6	21.46	[7.12,10.74]

table show that there is a significant improvement in terms of efficiency when implementing the configurations identified by the P^3-DSS. These results confirm the usefulness of the P^3-DSS as a policing DSS for the data and the parameters considered in the experiments.

Figures 6.10 - 6.12 show the best patrolling configurations obtained by the P^3-DSS for the three shifts considered. We can see that these patrol sectors have significant differences from those currently in use (Figure 6.7). Some insights will be given in the following:

- SATT3: Police activity is focused on the *Gran Via*, the main artery of the city that runs from the top-left corner of the district to the center, and then goes toward the east. The reason for that is that the *Gran Via* and its surroundings are very popular nightlife areas.

- SUNT1: The patrolling configuration concentrates on the southern part of the district, where most of the crimes happen on Sunday morning because of the popular flea market.

- MONT2: The city is uniformly partitioned between north-east and south-west. The configuration with 6 patrol sectors assigns higher importance to the central-western part of the district, corresponding to the commercial area.

Figure 6.10. Best patrolling configurations identified by the P^3-DSS for the shift SATT3 with 2 and 6 patrol sectors. Each sector is represented in a different color.

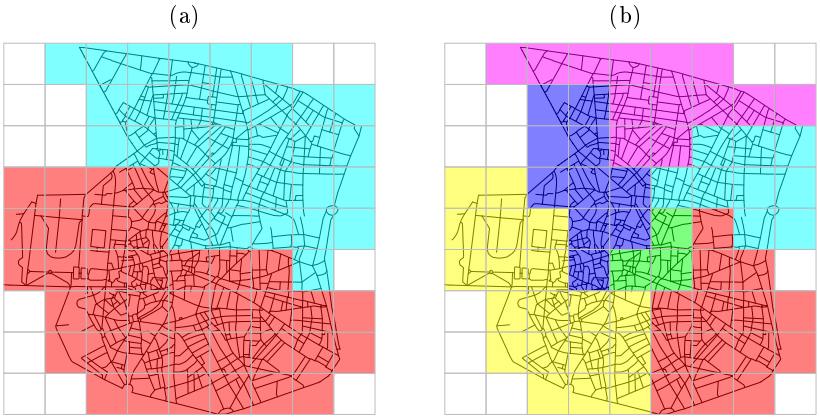
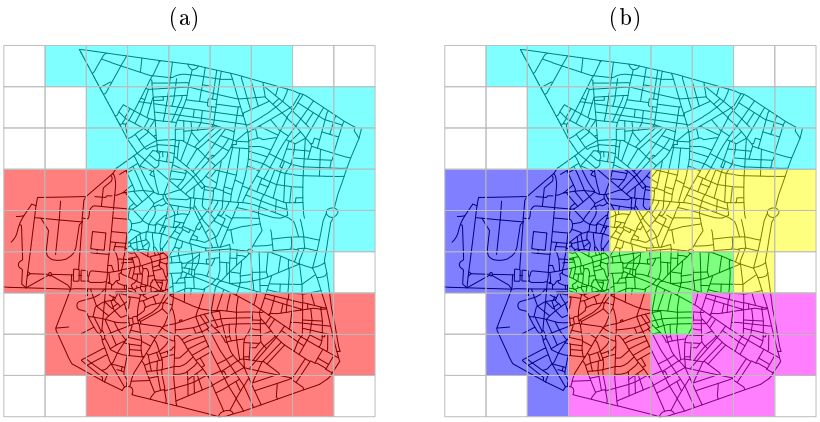


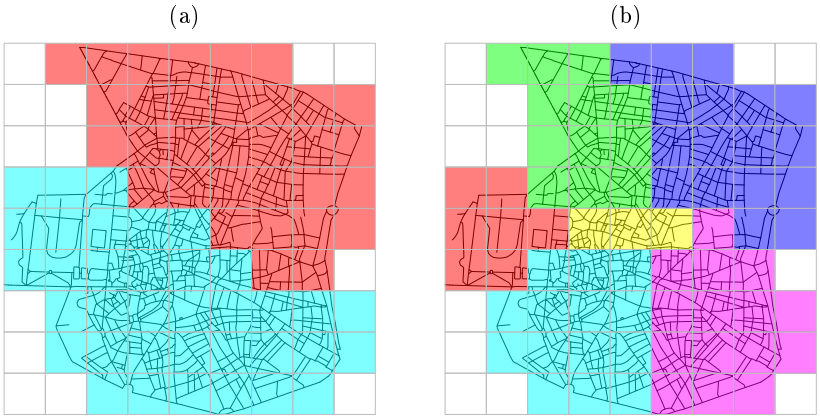
Figure 6.11. Best patrolling configurations identified by the P^3-DSS for the shift SUNT1 with 2 and 6 patrol sectors. Each sector is represented in a different color.



6.3. Local Search Methods for the MC-PDP on Graph Empirically Tested on a Case Study on the Central District of Madrid

We test our algorithm on the Central District of Madrid dataset presented in Camacho-Collados et al. [17]. However, in this research the data is aggregated with respect to the census district rather than a grid. As reported by Sarac et al. [89] the use of a structure based on census districts is preferable as it allows easy access to demographic data and is suitable for use by other agencies. Figure

Figure 6.12. Best patrolling configurations identified by the P^3-DSS for the shift MONT2 with 2 and 6 patrol sectors. Each sector is represented in a different color.



6.13 shows the subdivision of the territory and the associated graph. The borders of the census districts are plotted in gray. The nodes of the graph, identified by black bullets, correspond to the centroids of the census districts. Finally, black lines represent the edges of the graph that connect neighboring census districts. Overall, the graph is comprised of 111 nodes and 277 edges. The total length of the streets at each node, a_i, is obtained by summing the length of the parts of street contained within the borders of each census district. The length of each edge, l_{ij}, is computed as the great-circle distance between the nodes. In terms of the risk of crime at each node, r_i, we consider the thefts occurred during the following shifts:

- SATT3: Saturday, 10/13/2012, night shift (10 PM–8 AM).
- SUNT1: Sunday, 10/14/2012, morning shift (8 AM–3 PM).
- MONT2: Monday, 10/15/2012, afternoon shift (3 PM–10 PM).

These shifts have been identified by a service coordinator in charge of the patrolling operations of the Central District of Madrid as typical scenarios representing different crime activity patterns, as illustrated in Figure 6.14. In the SATT3 shift the district is characterized by a high level of nightlife, therefore thefts are committed in almost all the territory, with the highest levels distributed around popular meeting places in the center and in the north-east of the district. SUNT1 has a low level of criminality, mostly concentrated in the south of the district where a popular flea market is held every Sunday morning. Finally, MONT2 presents

Figure 6.13. Census districts in the Central District of Madrid (in gray) and the corresponding graph (in black).

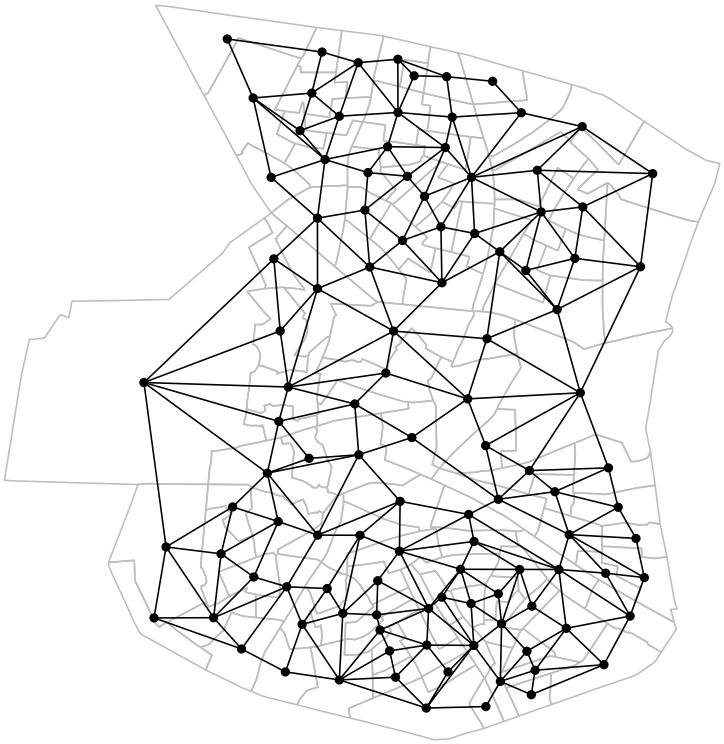
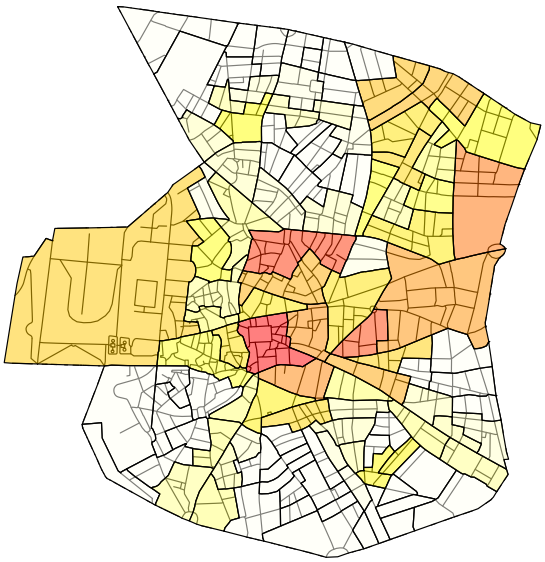
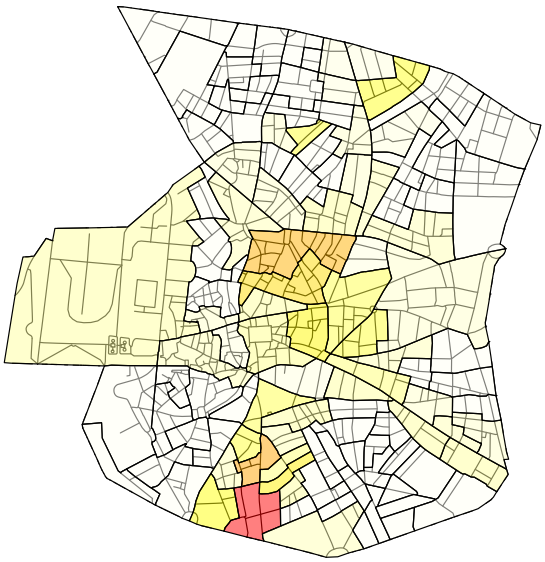


Figure 6.14. Maps of the number of thefts reported in the Central District of Madrid. The red shade represents a high crime level while the white shade represents no criminal activity.

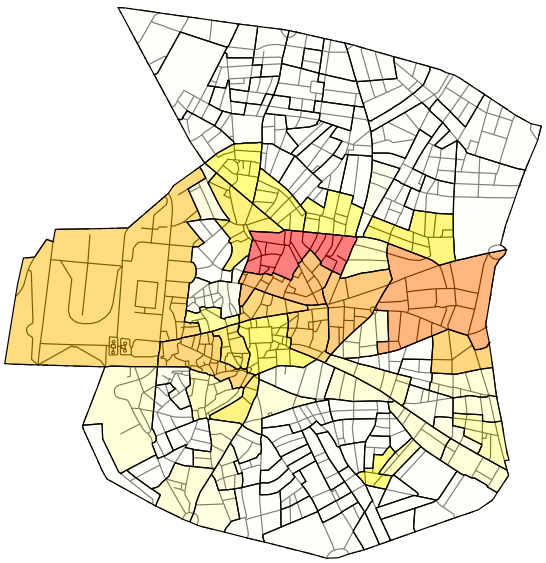
(a) SATT3



(b) SUNT1



(c) MONT2



the characteristics of a normal business day, with criminal activity spread in the central area of the territory, that is where the commercial activities are located.

6.3.1. Computational Experiments

The experiments were conducted using the same parameters adopted in previous researches on the subject [16, 17].

- Decision maker preference weights and balance coefficient: $(w_\alpha, w_\beta, w_\gamma, w_\delta) = (0.45, 0.05, 0.45, 0.05)$ and $\lambda = 0.1$. These values have been provided by a service coordinator in charge of the patrolling operations of the Central District of Madrid as her preference.
- Number of patrol sectors: $p = \{2, 6\}$. On an "average day," the Central District of Madrid is either split into two big sectors or partitioned according to its six neighborhoods.

Given the random nature of the algorithms proposed, we ran each combination of algorithm, shift and number of patrol sectors 50 times. Each run had a time limit of 60 seconds to simulate the real time environment of a DSS. The experiments were run on a computer with an Intel Core i5-2500K CPU having four cores at 3.30GHz and 4GB RAM memory and the algorithm were programmed in C++.

Tables 6.5a - 6.5f show the average relaxed objective function value, $\overline{obj}\,(P)$, and the corresponding standard deviation for each group. In the tables, the rows correspond to the algorithm and the best average solution value is highlighted in bold. Please note that a solution value that is less than one indicates that the solution is feasible with respect to the convexity constraints (5.15). From the tables we can observe that on average the TS algorithm finds the best solution in four out of six groups and the SDHC in the remaining two groups.

6.3.2. Statistical Analysis

To understand if the differences in the means are statistically significant we run one-way ANOVA tests. The results are illustrated in Table 6.6. We highlighted in bold the rows of the groups where a significant difference was detected. We can immediately see that there is no significant difference in the groups where the SDHC algorithm was the best. We run post-hoc Tukey's tests to understand

Table 6.5. Average relaxed objective function value, $\overline{obj}(P)$, and standard deviation for each group.

(a) Shift SATT3, $p = 2$.

Algorithm	Avg.	St. Dev.
SHC	0.50109	0.00435
SDHC	**0.49997**	**0.00413**
TS	0.53567	0.19831

(b) Shift SATT3, $p = 6$.

Algorithm	Avg.	St. Dev.
SHC	0.20720	0.00342
SDHC	0.20498	0.00313
TS	**0.20146**	**0.00513**

(c) Shift SUNT1, $p = 2$.

Algorithm	Avg.	St. Dev.
SHC	0.50456	0.01000
SDHC	0.50651	0.01277
TS	**0.49101**	**0.00473**

(d) Shift SUNT1, $p = 6$.

Algorithm	Avg.	St. Dev.
SHC	0.20619	0.00315
SDHC	0.20594	0.00328
TS	**0.20161**	**0.00384**

(e) Shift MONT2, $p = 2$.

Algorithm	Avg.	St. Dev.
SHC	0.50381	0.00608
SDHC	**0.50067**	**0.00656**
TS	0.51948	0.14180

(f) Shift MONT2, $p = 6$.

Algorithm	Avg.	St. Dev.
SHC	0.20350	0.00469
SDHC	0.20336	0.00498
TS	**0.19729**	**0.00620**

Table 6.6. Results of the one-way ANOVA tests on the solution values.

Shift	p	$F(2, 147)$	$\Pr(> F)$
SATT3	2	1.57	0.212
SATT3	**6**	**26.2**	**1.85e-10**
SUNT1	**2**	**37.44**	**7.19e-14**
SUNT1	**6**	**28.16**	**4.43e-11**
MONT2	2	0.754	0.472
MONT2	**6**	**22.12**	**4.01e-09**

Table 6.7. Results of Tukey's test for each group.

(a) Shift SATT3, $p = 2$.

Pair	p-value
SHC-SDHC	0.99867
TS-SDHC	0.26697
TS-SHC	0.28945

(b) Shift SATT3, $p = 6$.

Pair	p-value
SHC-SDHC	**0.01698**
TS-SDHC	**6.09e-5**
TS-SHC	**<1e-7**

(c) Shift SUNT1, $p = 2$.

Pair	p-value
SHC-SDHC	0.57937
TS-SDHC	**<1e-7**
TS-SHC	**<1e-7**

(d) Shift SUNT1, $p = 6$.

Pair	p value
SHC-SDHC	0.93070
TS-SDHC	**<1e-7**
TS-SHC	**<1e-7**

(e) Shift MONT2, $p = 2$.

Pair	p value
SHC-SDHC	0.98003
TS-SDHC	0.48723
TS-SHC	0.60639

(f) Shift MONT2, $p = 6$.

Pair	p value
SHC-SDHC	0.99018
TS-SDHC	**2e-7**
TS-SHC	**1e-7**

more in detail which algorithm performs better for the solution of the MC-PDP. Tukey's test is a single-step multiple comparison procedure used to find means that are significantly different from each other and that is more suitable for multiple comparisons than doing a number of t-tests would be. The results are illustrated in Tables 6.7a - 6.7f. In the tables, the rows are associated with the pairs of algorithms being tested. We highlighted in bold the rows showing a significant difference. From the results of the statistical tests we can draw the following conclusions:

- The performances of the SHC and the SDHC in terms of solutions objective function values are always identical, except for the shift SATT3 with six patrol sectors, where the SDHC produces solutions that are significantly better than those of the SHC.

- The TS produces on average better solutions in four out of six groups, and its performances are not worse than those of the other two algorithms in the remaining two groups. Therefore, we can claim that it is preferable to use the TS over the SHC and the SDHC.

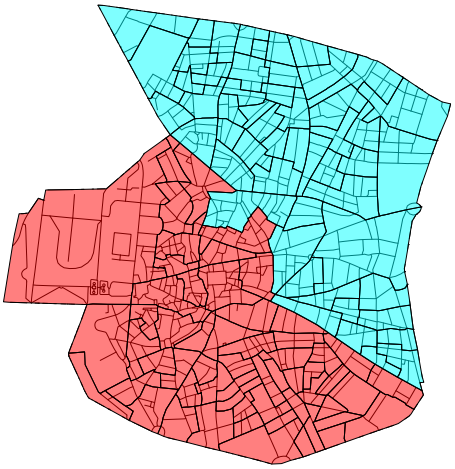
6.3.3. Solution Analysis

Figures 6.15a - 6.15f illustrate the best solutions found for each shift and number of patrol sectors in terms of relaxed objective function values. All the solutions have been identified by the TS. In the figures, the borders of the census districts have been plotted in black, the streets in gray and each patrol sector is represented by a different color. By observing the patrolling configurations some insights can be drawn:

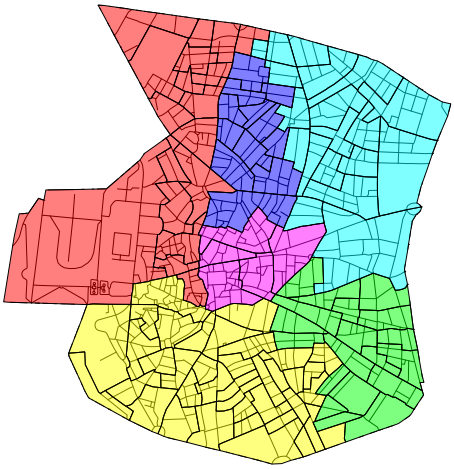
- SATT3: Police activity is focused mostly on the center, as well as on the north-east part of the district, where most of the crimes are committed. The reason for that is that those areas are very busy nightlife meeting places.
- SUNT1: The patrolling configurations concentrate on the southern part of the territory, where most of the thefts happen on Sunday morning because of the popular flea market. In the six patrol sectors configuration we can see that one sector is dedicated exclusively to the area with the highest concentration of crimes that corresponds exactly to the location of the flea market.
- MONT2: The district is uniformly partitioned between north-east and south-west. The configuration with six patrol sectors assigns higher importance to the central-western part of the district, corresponding to the commercial area.

Figure 6.15. Best solutions found.

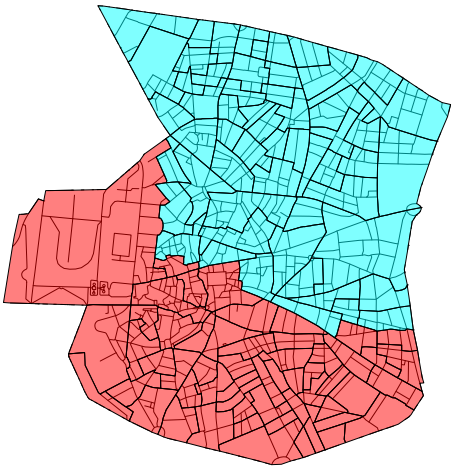
(a) Shift SATT3, $p = 2$.



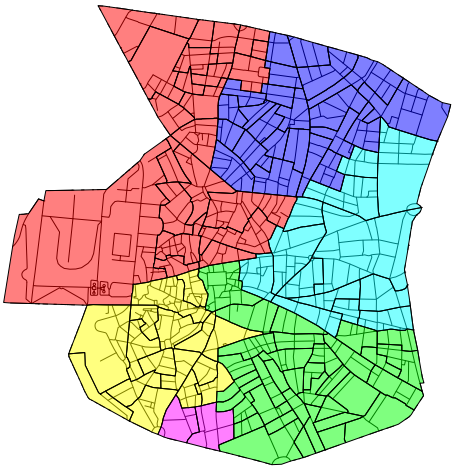
(b) Shift SATT3, $p = 6$.



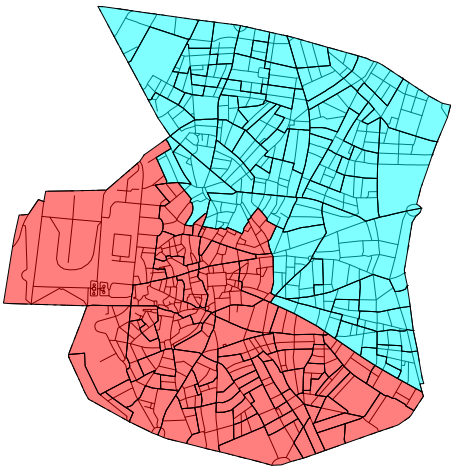
(c) Shift SUNT1, $p = 2$.



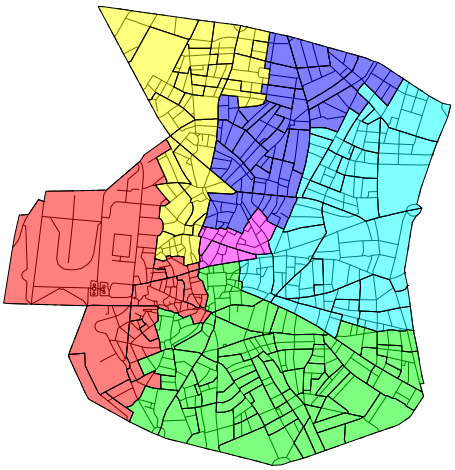
(d) Shift SUNT1, $p = 6$.



(e) Shift MONT2, $p = 2$.



(f) Shift MONT2, $p = 6$.



CHAPTER 7

CONCLUSIONS AND FUTURE RESEARCH LINES

In this thesis the author presented a new model for the improvement of police district design by homogenous distribution of workload among the sectors that comprise a district. In this model, specifically tailored to suit the requirements of the SNPC, the workload for each sector is measured in terms of total associated area, risk, diameter (that is, a proxy for compactness), and mutual support. The area is obtained from GIS maps and the risk from a prediction of the total number of crimes in the district. Even distribution of the workload among the sectors improves the efficency of security operations and the satisfaction of the agents.

The first original contribution of the thesis concerned the extension of the existing PDP models to include features that match the requirements of the SNPC. In particular, no previous model included a measure of mutual support that allows for considering the effect of being able to rely on the assistance of neighboring sectors.

The second contribution concerned the adaptation of the model on a generic graph. The main challenge was represented by the convexity constraints on the sectors. Convex partitioning of graphs is a very recent field of research that is still at its infancy. In this thesis, the author proposed a novel convexity condition for graphs that presents advantages in terms of computational cost.

Given the non-linear nature of the model restrictions, the author explored local search heuristics for its solution. Specifically, the author compared a SHC, a SDHC, and a TS. A case study of the Central District of Madrid including real criminal data was presented and the performance of the algorithms was assessed. The author showed empirically that all the algorithms rapidly generates patrolling configurations that are more efficient than those currently adopted by the SNPC. More in detail, the TS provided the best solutions among the three algorithms.

The research presented in this thesis fosters a number of new research directions and several areas that can be potentially extended as described below. The approximation introduced by the current area measure could be improved by considering other more realistic measures. A previous implementation of the model computed the minimum length Hamiltonian Cycle. However, preliminary computational experiments showed that that was computationally inefficient. Further research could focus on its time-efficient implementation, or on alternative representative measures [76, 77].

Furthermore, the effectiveness of other heuristic and metaheuristic algorithms such as ant colonies and genetic algorithms could be investigated. The solution by means of optimal methods would open new research opportunities for the presented model, such as the analysis of the inclusion of social factors among the criteria considered. Although the model is intrinsically non-linear, decomposition methods such as Column Generation or Benders' Decomposition could be applied to solve the problem to optimality. Also, these methodologies could still be used to generate good heuristic solutions should the solution process take longer than the allowed computational time.

Recent papers have analyzed the statistical effect of law enforcement actions on crime patterns [52]. By including these effects in an optimization problem it would be possible to formulate a model for the design of patrol configurations that result in a reduction of the future level of criminality. The model would be similar in nature to theoretical games [48] and to fortification/interdiction problems used to hedge against intentional attacks [60, 91, 116] and natural disasters [65].

From the point of view of spatio-temporal risk assessment, an important direction of research concerns the implementation in this context of different risk measures for objetive model-based dynamical specification of risk inputs in the DSS.

On a different note, a service coordinator in charge of the patrolling operations in the Central District of Madrid pointed out that an important component is ensuring that the agents' job is "pleasant", as opposed to dull and boring. It could be an interesting challenge for modelers to come up with an "interesting" attribute to be included during the optimization process.

The author wishes that this thesis will be a useful source of inspiration for future research on police districting problems, and will contribute further to the development of solution approaches that can solve more complex and realistic

models within the context of public security.

CHAPTER 7

CONCLUSIONES Y FUTURAS LÍNEAS DE INVESTIGACIÓN

En esta tesis el autor presenta un nuevo modelo para la mejora del diseño de la policía distrital a través de una distribución homogénea de la carga de trabajo entre los sectores que componen un distrito. En este modelo, adaptado específicamente para satisfacer los requisitos del CNP, la carga de trabajo para cada sector se mide en términos de la superficie total de área asociada, el riesgo, el diámetro (es decir, un indicador de compactibilidad), y el apoyo mutuo. El área se obtiene a partir de mapas SIG y el riesgo de una predicción del número total de delitos en el distrito. Incluso la distribución de la carga de trabajo entre los sectores mejora la eficiencia de las operaciones de seguridad y la satisfacción de los agentes.

La primera contribución original de la tesis se refiere a la extensión de los modelos PDDP existentes para incluir características que responden a las exigencias del CNP. En particular, ningún modelo anterior incluía una medida de apoyo mutuo que permitiese considerar el efecto de poder contar con la ayuda de los sectores vecinos.

La segunda contribución versa sobre la adaptación del modelo en un grafo genérico. El reto principal fue representado por las limitaciones de convexidad en los sectores. La división convexa de los grafos es un campo muy reciente de la investigación que se encuentra todavía en su etapa más temprana. En esta tesis, el autor propone una condición de convexidad novedosa para grafos que presenta ventajas en términos de coste computacional.

Dada la naturaleza no lineal de las restricciones del modelo, el autor explora los heurísticos de búsqueda local para su solución. En concreto, el autor comparó un SHC, un SDHC y un BT. Se presenta un caso de estudio en el Distrito Central de Madrid y el rendimiento del algoritmo es evaluado. El autor demuestra empíri-

camente que el algoritmo genera rápidamente configuraciones de patrulla que son más eficientes que las actualmente empleadas por el CNP. Más en detalle, el BT proporciona las mejores soluciones entre los tres algoritmos.

La investigación presentada en esta tesis fomenta una serie de nuevas direcciones de investigación y varias áreas que pueden ser potencialmente ampliadas como se describe a continuación. La aproximación introducida por la medida de área actual podría mejorarse teniendo en cuenta otras medidas más realistas. Una aplicación anterior del modelo calcula la longitud mínima del camino hamiltoniano. Sin embargo, los experimentos computacionales preliminares mostraron que éste era computacionalmente ineficiente. La investigación adicional podría centrarse en su aplicación tiempo-eficiencia, o en medidas alternativas de representación [76, 77].

Desde el punto de vista de la valoración espacio-temporal del riesgo, una importante dirección de investigación concierne la implementación en este contexto de diferentes medidas para una especificación dínamica y objetiva del riesgo para el SSD basada en modelos.

Por otra parte, la efectividad de otros algoritmos heurísticos y meta heurísticos tales como las colonias de hormigas y algoritmos genéticos podrían ser investigadas. La solución por medio de métodos óptimos abriría nuevas oportunidades de investigación para el modelo presentado, tales como el análisis de la inclusión de factores sociales entre los criterios considerados. Aunque el modelo es intrínsecamente no lineal, métodos de descomposición, tales como la Generación de Columnas o de Descomposición de Bender podrían aplicarse para resolver el problema de optimización. Además, estas metodologías todavía se podrían utilizar para generar buenas soluciones heurísticas en caso de que el proceso de solución llevara más tiempo que el tiempo de cálculo permitido.

Trabajos recientes han analizado el efecto estadístico de las acciones de la aplicación de la ley sobre los patrones de delitos [52]. Mediante la inclusión de estos efectos en un problema de optimización, sería posible formular un modelo para el diseño de configuraciones de patrullaje que resulten en una reducción del futuro nivel de criminalidad. El modelo sería de naturaleza similar a los juegos teóricos [48] y para la fortificación e interdicción de problemas utilizados para protegerse contra los ataques intencionales [60, 91, 116] y los desastres naturales [65].

En otro orden de ideas, un coordinador de servicios a cargo de las operaciones de patrullaje en el Distrito Central de Madrid señaló que un componente

importante es asegurarse de que el trabajo de los agentes sea agradable, en lugar de gris y aburrido. Podría ser un reto interesante para los diseñadores llegar a obtener un atributo interesante para ser incluidos en el proceso de optimización.

El autor desea que esta tesis sea una fuente útil de inspiración para futuras investigaciones sobre los problemas de los distritos policiales, y siga contribuyendo al desarrollo de soluciones que puedan resolver modelos más complejos y realistas dentro del contexto de la seguridad pública.

MAIN CONSTRIBUTIONS AND MERITS

In this chapter the contributions and the main merits of this research are presented.

a) Journal Publications

- M. Camacho-Collados, F. Liberatore, and J.M. Angulo, "A multi-criteria police districting problem for the efficient and effective design of patrol sector", European Journal of Operational Research 246(2):674–684, 2015. JCR rank: 9/82, Q1, Operations Research & Management Science.
- M. Camacho-Collados and F. Liberatore, "A decision support system for predictive police patrolling", Decision Support Systems 75:25–37, 2015. JCR rank: 10/82, Q1, Operations Research & Management Science.
- F. Liberatore and M. Camacho-Collados, "A comparison of local search methods for the multicriteria police districting problem on graph", Mathematical Problems in Engineering, Vol. 2016, 2016. JCR rank: 81/101, Q4, Mathematics, Interdisciplinary Applications.

b) Conferences, Symposiums and Talks

- M. Camacho-Collados and F. Liberatore, "La lucha contra el crimen a través de Métodos Estadísticos: presente y futuro", Segundo Ciclo de Conferencias del Año Internacional de la Estadística, Universidad de Granada, 01/20/2013.
- M. Camacho-Collados, round table "Crime prediction and its application on the Civil Society", Security Forum 2014, 05/28/2014.

- M. Camacho-Collados, "Spatial Partitioning of Police Districts: a Multi-Criteria Model", Forum on Spatial Thinking, University of California Santa Barbara, 07/15/2015.
- M. Camacho-Collados and F. Liberatore, "Modelos estadísticos policiales para un uso eficiente de los medios, materiales y humanos, encaminados a una prevención eficaz del delito", Universidad Internacional de Valencia, 12/10/2015.
- M. Camacho-Collados and F. Liberatore, "Herramientas de partición espacial de distritos policiales a través de programas estadísticos", Universidad Europea de Valencia, 12/10/2015.
- M. Camacho-Collados, round table "Oportunidades derivadas de la relación industria- academia. Administración pública en el ámbito de la transferencia de tecnología matemática", Jornada CTA- IMUS-MathIn de transferencia tecnológica matemática, Universidad de Sevilla, 11/12/2015.
- M. Camacho-Collados, "Modelos matemáticos predictivos de riesgo criminal", Jornada CTA- IMUS-MathIn de transferencia tecnológica matemática, Universidad de Sevilla, 11/12/2015.
- M. Camacho-Collados, "Spatiotemporal patterns of crime in Smart Patrolling", Cátedra Eurocop de Predicción del Delito, Universitat Jaume I de Castellón, 02/19/2016.
- M. Camacho-Collados, "Modelos estadísticos policiales para una prevención eficaz del delito". I Jornadas de Investigadores en Formación: fomentando la interdisciplinariedad (JIFFI), Universidad de Granada, 05/20/2016.
- F. Liberatore and M. Camacho-Collados, "Multi-Criteria Police Districting Problem (PDP)", X Reunión del Grupo Español de Decisión Multicriterio, Universidad CEU San Pablo, 06/10/2016.
- M. Camacho-Collados, F. Liberatore, and J.M. Angulo, "A spatial partitioning of Police Districts", 26th Annual Conference of the International Environmetrics Society, University of Edinburgh, 07/20/2016.

c) Poster Presentations

- M. Camacho-Collados, F. Liberatore, and J.M. Angulo, "A multi-criteria police districting problem for the efficient and effective design of patrol sector", The 3rd Annual Review of ARO Game Theory MURI, University of Southern California, Los Angeles (CA), USA, 12/05/2014.
- M. Camacho-Collados, F. Liberatore, and J.M. Angulo, "A multi-criteria police districting problem for the efficient and effective design of patrol sector", AFOSR-MURI meeting, Marina del Rey (CA), USA, 09/22/2014.
- M. Camacho-Collados, F. Liberatore, and J.M. Angulo, "A multi-criteria police districting problem for the efficient and effective design of patrol sector", NGA Symposium, Washington (DC), USA, 09/08/2014.

d) Research Visits

- Applied Mathematics research group, Department of Mathematics, University of California Los Angeles, Los Angeles (CA), USA, 09/01/2014–08/31/2015. Supervisor: Prof. Andrea L. Bertozzi.

e) Grants/Scholarships

- Fulbright Foundation, Fulbright scolarship for doctoral studies in the United States. Awarded June 2013.
- Spanish Police Foundation, Grant for research and elaboration of police studies. Awarded July 2013.

f) Research Projects

- Title: Risk Analysis in Complex Systems. Theoretical and Methodological Advances.

Reference: MTM2015-70840-P.
Principal Investigator: José Miguel Angulo.
Founding Entity: Ministerio de Economía y Competitividad. Secretaría de Estado de Investigación, Desarrollo e Innovación. Dirección General de Investigación Científica y Técnica. Subdirección General de Proyectos de Investigación.
Call: Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento, convocatoria 2015, modalidad 1: Proyectos de I+D.
Beneficiary: Facultad de Ciencias, Universidad de Granada.
Duration: 01/01/2016 – 12/31/2018.
Amount: 18,150.00 €.

g) Patents and Industrial / Intellectual Properties

- Title: Herramienta de patrullaje inteligente.
  Authors: Miguel Camacho Collados, Federico Liberatore.
  Reference: 16/2014/1523.
  Country: Spain, Community of Madrid.
  Date: 01/24/2014.

h) Media Impact

- "Desarrollan un sistema informático rápido y barato que permite predecir los delitos y organizar mejor los turnos policiales", Secretaría General, Universidad de Granada, 10/01/2015.
- "Diseñan un sistema para predecir delitos y organizar mejor turnos policiales", Europa Press, 10/01/2015.
- "Sistema informático permite predecir delitos y organizar mejor turnos Policía", El Confidencial, 10/01/2015.

- "Policía Nacional y UGR desarrollan un sistema para predecir delitos y organizar mejor los turnos policiales", 20 minutos, 10/01/2015.
- "Big Data para predecir delitos", Catalunya Vanguardista, 10/01/2015.
- "El Big Data, «apatrullando» la ciudad", La Razón, 10/02/2015.
- "Una app para predecir los delitos", Quo, 10/02/2015.
- B. González de Vega, "El 'poli' granadino que usa las matemáticas contra los malos", El Mundo, 11/14/2015.
- S. Allocca, "Can Big Data Help Predict Crime?", Forensic Magazine, 03/08/2016.

REFERENCES

1. R. Benveniste. Solving the combined zoning and location problem for several emergency units. *Journal of the Operational Research Society*, 36(5):433–450, 1985.
2. P. Bergey, C. Ragsdale, and M. Hoskote. A decision support system for the electrical power districting problem. *Decision Support Systems*, 36(1):1–17, 2003.
3. P. Bergey, C. Ragsdale, and M. Hoskote. A simulated annealing genetic algorithm for the electrical power districting problem. *Annals of Operations Research*, 121(1-4):33–55, 2003.
4. R. Bivand and N. Lewin-Koh. *maptools: Tools for reading and handling spatial objects*, 2014. URL http://CRAN.R-project.org/package=maptools. R package version 0.8-29.
5. R. Bivand and C. Rundel. *rgeos: Interface to Geometry Engine - Open Source (GEOS)*, 2014. URL http://CRAN.R-project.org/package=rgeos. R package version 0.3-4.
6. R.S. Bivand, E. Pebesma, and V. Gomez-Rubio. *Applied spatial data analysis with R*. Springer-Verlag, 2013. URL http://www.asdar-book.org/. Second edition.
7. M. Blais, S. Lapierre, and G. Laporte. Solving a home-care districting problem in an urban setting. *Journal of the Operational Research Society*, 54(11): 1141–1147, 2003.
8. S. Bodily. Police sector design incorporating preferences of interest groups for equality and efficiency. *Management Science*, 24(12):1301–1313, 1978.
9. B. Bozcaya, E. Erkut, and G. Laporte. A tabu search heuristic and adaptive memory procedure for political districting. *European Journal of Operational Research*, 144(1):12–26, 2003.
10. J. Braa and C. Hedberg. The struggle for district-based health information

systems in South Africa. *Information Society*, 18(2):113–127, 2002.

11. P.J. Brantingham and P. Brantingham. *Patterns in crime*. Prentice-Hall, 1984.
12. M.M. Brown. The benefits and costs of information technology innovations: An empirical assessment of a local government agency. *Public Performance and Management Review*, 24(4):351–366, 2001.
13. M.M. Brown and J.L. Brudney. Learning organizations in the public sector? a study of police agencies employing information technology to advance knowledge. *Public Administration Review*, 63:30–43, 2003.
14. C. Bruce. Districting and resource allocation: A question of balance. *Geography & Public Safety*, 1(4):1–3, 2009.
15. R.J. Bursik and H.G. Grasmick. *Neighborhoods and crime: The dimensions of effective community control*. Lexington Books, 1993.
16. M. Camacho-Collados and F. Liberatore. A decision support system for predictive police patrolling. *Decision Support Systems*, 75:25–37, 2015.
17. M. Camacho-Collados, F. Liberatore, and J.M. Angulo. A multi-criteria police districting problem for the efficient and effective design of patrol sector. *European Journal of Operational Research*, 246:674–684, 2015.
18. J.M. Caplan and L.W. Kennedy. *Risk Terrain Modeling Compendium*. Rutgers Center on Public Security, 2011.
19. J.M. Caplan, L.W. Kennedy, and J. Miller. Risk terrain modeling: Brokering criminological theory and GIS methods for crime forecasting. *Justice Quarterly*, 28(2):360–381, 2011.
20. F. Caro, T. Shirabe, M. Guignard, and A. Weintraub. School redistricting: Embedding GIS tools with integer programming. *Journal of the Operational Research Society*, 55(8):836–849, 2004.
21. J. Chaiken and P. Dormont. A patrol car allocation model: Background. *Management Science*, 24(12):1280–1290, 1978.
22. J. Chaiken and P. Dormont. A patrol car allocation model: Capabilities and algorithms. *Management Science*, 24(12):1291–1300, 1978.
23. J. Chan. The technology game: How information technology is transforming police practice. *Journal of Criminal Justice*, 1:139–159, 2001.
24. S. Chaturapruek, J. Breslau, D. Yazdi, T. Kolokolnikov, and S.G. McCalla. Crime modeling with Levy flights. *SIAM Journal on Applied Mathematics*, 73(4):1703–1720, 2013.

25. P. Chen, H. Yuan, and D. Li. Space-time analysis of burglary in Beijing. *Security Journal*, 26(1):1–15, 2013.
26. R.L. Church and M.P. Scaparra. Analysis of facility systems reliability when subject to attack or a natural disaster. In A.T. Murray and T.H. Grubesic, editors, *Critical infrastructure: Reliability and vulnerability: Advances in spatial science*, pages 221–241. Springer-Verlag, 2007.
27. C. Cirincione, T. Darling, and T. O'Rourke. Assessing South Carolina's 1990s congressional districting. *Political Geography*, 19(2):189–211, 2000.
28. J. Cohen. Development of crime forecasting and mapping systems for use by police. Technical report, U.S. Department of Justice, 2006.
29. K. Curtin, F. Qui, K. Hayslett-McCall, and T. Bray. *Geographic Information Systems and Crime Analysis*, chapter Integrating GIS and maximal coverage models to determine optimal police patrol areas, pages 214–235. Idea Group Inc., 2005.
30. K. Curtin, K. Hayslett-McCall, and F. Qiu. Determining optimal police patrol areas with maximal covering and backup covering location models. *Networks and Spatial Economics*, 10(1):125–145, 2010.
31. S. D'Amico, S. Wang, R. Batta, and C. Rump. A simulated annealing approach to police district design. *Computers & Operations Research*, 29(6): 667–684, 2002.
32. J. Douglass. Tactical deployment: The next great paradigm shift in law enforcement? *Geography & Public Safety*, 1(4):6–7, 2009.
33. A. Drexl and K. Haase. Fast approximation methods for sales force deployment. *Management Science*, 45(10:1307–1323, 1999.
34. R.W. Floyd. Algorithm 97: Shortest path. *Communications of the ACM*, 5(6):345, 1962.
35. Z. Friend. Predictive policing: Using technology to reduce crime. FBI Law Enforcement Bulletin, 2013.
36. V. Furtado, A. Melo, A.L.V. Coelho nd R. Menezes, and R. Perrone. A bio-inspired crime simulation model. *Decision Support Systems*, 48(1):282–292, 2009.
37. L. Galvao, A. Novaes, J. Souza de Cursi, and J. Souza. A multiplicatively-weighted Voronoi diagram approach to logistics districting. *Computers & Operations Research*, 33(1):93–114, 2006.
38. M.S. Gerber. Predicting crime using Twitter and kernel density estimation.

Decision Support Systems, 61(1):115–125, 2014.

39. F. Glover. Tabu search - part i. *ORSA Journal on Computing*, 1(3):190–206, 1989.
40. F. Glover. Tabu search - part ii. *ORSA Journal on Computing*, 2(1):4–32, 1990.
41. W. Gorr and R. Harries. Introduction to crime forecasting. *International Journal of Forecasting*, 19(4):551–555, 2003.
42. W. Gorr, A. Olligschlaeger, and Y. Thompson. Short-term forecasting of crime. *International Journal of Forecasting*, 19(4):579–594, 2003.
43. E. Groff and T. McEwen. *Identifying and Measuring the Effects of Information Technologies on Law Enforcement Agencies*. Penny Hill Press Inc, 2008.
44. S. Hanafi and A.V. Freville. Municipal solid waste collection: An effective data structure for solving the sectorization problem with local search methods. *INFOR*, 37(3):236–254, 1999.
45. C. Harris. *The New technology of Crime, Law and Social Control*, chapter Police and Soft Technology: How Information Technology Contributes to Police Decision Making, pages 153–183. Criminal Justice Press, 2007.
46. D. He and Y.L. Hong. An improved tabu search algorithm based on grid search used in the antenna parameters optimization. *Mathematical Problems in Engineering*, 2015:8, 2015. Article ID 947021.
47. S. Hess, J. Weaver, H. Siegfeldt, J. Whelan, and P. Zitlau. Non-partisan political redistricting by computer. *Operations Research*, 13(6):998–1008, 1965.
48. R. Hohzaki and H. Maehara. A single-shot game of multi-period inspection. *European Journal of Operational Research*, 207(3):1410–1418, 2010.
49. R.J. Hyndman. *forecast: Forecasting functions for time series and linear models*, 2014. URL http://CRAN.R-project.org/package=forecast. R package version 5.4.
50. R.J. Hyndman, A.B. Koehler, J.K. Ord, and R.D. Snyder. *Forecasting with Exponential Smoothing: The State Space Approach*. Springer-Verlag, 2008.
51. A.P. Iannoni, R. Morabito, and C. Saydam. An optimization approach for ambulance location and the districting of the response segments on highways. *European Journal of Operational Research*, 195(2):528–542, 2009.
52. P.A. Jones, P.J. Brantingham, and L.R. Chayes. Statistical models of criminal behavior: The effects of law enforcement actions. *Mathematical Models and*

Methods in Applied Sciences, 20, Suppl.:1397–1423, 2010.

53. J.N. Kalcsics and M. Schröeder. Towards a unified territorial design approach - applications, algorithms and GIS integration. *TOP*, 13(1):1–74, 2005.
54. N. Karma, C.Y. Suen, and P.F. Guo. Palmprints: A novel co-evolutionary algorithm for clustering finger images. In *Genetic and Evolutionary Computation - GECCO 2003*, volume 2723 of *Lecture Notes in Computer Science*, pages 322–331. Springer-Verlag, 2003.
55. L.W. Kennedy, J.M. Caplan, and E. Piza. Risk clusters, hotspots, and spatial intelligence: Risk terrain modeling as an algorithm for police resource allocation strategies. *Journal of Quantitative Criminology*, 27(3):339–362, 2011.
56. H. Kim and M. O'Kelly. Survivability of commercial backbones with peering: A case study of Korean networks. In A.T. Murray and T.H. Grubesic, editors, *51st annual North American meetings of the Regional Science Association International*, pages 107–128. Springer-Verlag, 2004.
57. D.M. King, S.H. Jacobson, E.C. Sewell, and W.K. Tam Cho. Geo-graphs: An efficient model for enforcing contiguity and hole constraints in planar graph partitioning. *Operations Research*, 60(5):1213–1228, 2012.
58. D.M. King, S.H. Jacobson, and E.C. Sewell. Efficient geo-graph contiguity and hole algorithms for geographic zoning and dynamic plane graph partitioning. *Mathematical Programming*, Published online, 2014.
59. A. Kistler. Tucson police officers redraw division boundaries to balance their workload. *Geography & Public Safety*, 1(4):3–5, 2009.
60. M. Kress, J.O. Royset, and N. Rozen. The eye and the fist: Optimizing search and interdiction. *European Journal of Operational Research*, 220(2):550–558, 2012.
61. P.F. Kuo, D. Lord, and T.D. Walden. Using geographical information systems to organize police patrol routes effectively by grouping hotspots of crash and crime data. *Journal of Transport Geography*, 30(1):138–148, 2013.
62. R. Larson. A hypercube queuing model for facility location and redistricting in urban emergency services. *Computers & Operations Research*, 1(1):67–95, 1974.
63. S. Li, S. Kuo, and F. Tsai. An intelligent decision-support model using FSOM and rule extraction for crime prevention. *Expert Systems with Applications*, 37(10):7108–7119, 2010.

64. F. Liberatore and M. Camacho-Collados. A comparison of local search methods for the multicriteria police districting problem on graph. *Mathematical Problems in Engineering*, 2016:13 pages, 2016.
65. F. Liberatore, M.P. Scaparra, and M.S. Daskin. Hedging against disruptions with ripple effects in location analysis. *Omega*, 40 (1):21–30, 2012.
66. G. Lin, W.X. Zhu, and M.M. Ali. A tabu search-based memetic algorithm for hardware/software partitioning. *Mathematical Problems in Engineering*, 2014:15, 2014. Article ID 103059.
67. A. Mayer. Geospatial technology helps east orange crack down on crime. *Geography & Public Safety*, 1(4):8–10, 2009.
68. A. Mehrotra, E. Johnson, and G. Nemhauser. An optimization based heuristic for political districting. *Management Science*, 44(8):1100–1114, 1998.
69. P. Mielke. Using ESRI's districting tool in policing. *Geography & Public Safety*, 1(4):10–13, 2009.
70. R.P. Minciardi and R. Zoppoli. A districting procedure for social organizations. *European Journal of Operational Research*, 8(1):47–57, 1981.
71. P.S. Mitchell. Optimal selection of police patrol beats. *The Journal of Criminal Law, Criminology and Police Science*, 63(4):577, 1972.
72. G. Mohler, M. Short, P. Brantingham, F. Schoenberg, and G. Tita. Self-exciting point process modeling of crime. *Journal of the American Statistical Association*, 106(493)(100-108), 2011.
73. L. Muyldermans. Routing, districting and location for arc traversal problems. *4OR*, 1(2):169–172, 2003.
74. L. Muyldermans, D. Cattrysse, D. Van Oudheusden, and T. Lotan. Districting for salt spreading operations. *European Journal of Operational Research*, 139(3):521–532, 2002.
75. NIJ. Predictive policing, 2014. http://www.nij.gov/topics/law-enforcement/strategies/predictive-policing/Pages/welcome.aspx.
76. S. Opasanon and E. Miller-Hooks. Multicriteria adaptive paths in stochastic, time-varying networks. *European Journal of Operational Research*, 173(1): 72–91, 2006.
77. R. Pal and I. Bose. An optimization based approach for deployment of roadway incident response vehicles with reliability constraints. *European Journal of Operational Research*, 198(2):452–463, 2009.
78. K. Park, K. Lee, S. Park, and H. Lee. Telecommunication node clustering

with node compatibility and network survivability requirements. *Management Science*, 46(3):363–374, 2000.

79. E.J. Pebesma and R.S. Bivand. Classes and methods for spatial data in R. *R News*, 5(2), 2005. URL http://cran.r-project.org/doc/Rnews/.
80. W.L. Perry, B. McInnis, C.C. Price, S.C. Smith, and J.S. Hollywood. *Predictive Policing. The Role of Crime Forecasting in Law Enforcement Operations*. RAND Corporation, 2013.
81. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014. URL http://www.R-project.org/.
82. J.H. Ratcliffe. Aoristic signatures and the spatio-temporal analysis of high volume crime patterns. *Journal of Quantitative Criminology*, 18(1):23–43, 2002.
83. K. Reichert. Use of information technology by law enforcement. In *Forum on Crime and Justice*, 2001.
84. D.J. Roberts. Technology is playing an expanding role in policing. *The Police Chief*, 78:72–73, 2011.
85. C. Romero. Extended lexicographic goal programming: A unifying approach. *Omega*, 29:63–71, 2001.
86. C. Romero. A general structure of achievement function for a goal programming model. *European Journal of Operational Research*, 153(1):675–686, 2004.
87. S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, 3rd edition edition, 2009.
88. M. Safe, J. Carbadillo, I. Ponzoni, and N. Brignole. On stopping criteria for genetic algorithms. In A.L.C. Bazzan and S. Labidi, editors, *Advances in Artificial Intelligence - SBIA 2004*, volume 3171 of *Lecture Notes in Computer Science*, pages 405–413. Springer-Verlag, 2004.
89. A. Sarac, R. Batta, J. Bhadury, and C. Rump. Reconfiguring police reporting districts in the city of Buffalo. *OR Insight*, 12(3):16–24, 1999.
90. N. Scalisi, J. Beres, S. Sharpe, R. Wilson, T. Brown, and A. Whitworth, editors. *Geography & Public Safety Bulletin*, volume 1(4). COPS and NIJ, 2009.
91. M.P. Scaparra and R.L. Church. An exact solution approach for the interdiction median problem with fortification. *European Journal of Operational*

Research, 189 (1):76–92, 2008.

92. O. Schoepfle and R. Church. A new network representation of a "classic" school districting problem. *Socio-Economic Planning Sciences*, 25(3):189–197, 1991.
93. P. Schultz. Future is here: Technology in police departments. *The Police Chief*, 75(6), 2008.
94. L.W. Sherman, P.R. Gartin, and M.E. Buerger. Hot spots of predatory crime: Routine activities and the criminology of place. *Criminology*, 27(1):27–56, 1989.
95. T. Shirabe. A model of contiguity for spatial unit allocation. *Geographical Analysis*, 37(1):2–16, 2005.
96. T. Shirabe. Districting modeling with exact contiguity constraints. *Environment and Planning B: Planning and Design*, 36(6):1053–1066, 2009.
97. M. Short, A. Bertozzi, and P.J. Brantingham. Nonlinear patterns in urban crime: Hotspots, bifurcations, and suppression. *SIAM Journal on Applied Dynamical Systems*, 9(2):462–483, 2010.
98. M.B. Short, M.R. D'Orsogna, V.B. Pasour, G.E. Tita, P.J. Brantingham, A.L. Bertozzi, and L.B. Chayes. A statistical model of criminal behavior. *Mathematical Models and Methods in Applied Sciences*, 18, Suppl.:1249–1267, 2008.
99. M.A. Smith and D.E. Brown. Application of discrete choice analysis to attack point patterns. *Information Systems and e-Business Management*, 5(3):255–274, 2007.
100. M. Stroshine. *Critical Issues in Policing: Contemporary Readings*, chapter Information technology innovations in policing. 2005.
101. F. Tavares-Pereira, J. Rui Figueira, V. Mousseaus, and B. Roy. Multiple criteria districting problems - The public transportation network pricing system of the Paris region. *Annals of Operations Research*, 154(1):69–92, 2007.
102. F. Tavares-Pereira, J. Rui Figueira, V. Mousseaus, and B. Roy. Comparing two territory partitions in districting problems:Indices and practical issues. *Socio-Economic Planning Sciences*, 43(1):72–88, 2009.
103. P.E. Taylor and S.J. Huxley. A break from tradition for the San Francisco police: Patrol officer scheduling using an optimization-based decision support system. *Interfaces*, 19(1):4–24, 1989.
104. D. Urban and T. Keitt. Landscape connectivity: A graph theoretic perspec-

tive. *Ecology*, 82(5):1205–1218, 2001.

105. F. Wang. Why police and policing need GIS: An overview. *Annals of GIS*, 18(3):159–171, 2012.
106. X. Wang, M.S. Gerber, and D.E. Brown. *"Social Computing, Behavioral - Cultural Modeling and Prediction"*, chapter Automatic Crime Prediction Using Events Extracted from Twitter Posts, pages 231–238. Springer-Verlag, 2012.
107. S. Warshall. A theorem on Boolean matrices. *Journal of the ACM*, 9(1): 11–12, 1962.
108. D. Weisburd and L. Green. Policing drug hot spots: The Jersey City drug market analysis experiment. *Justice Quarterly*, 12(4):711–36, 1995.
109. D. Weisburd, L. Maher, and L. Sherman. *Contrasting Crime General and Crime Specific Theory: The Case of Hot Spots of Crime*, volume 4 of *Advances in Criminological Theory*. Transaction Press, 1992.
110. D. Weisburd, S. Mastrofski, A.M. McNally, R. Greenspan, and J. Willis. Reforming to preserve: Compstat and strategic problem solving in American policing. *Criminology and Public Policy*, 2(3):421–456, 2003.
111. Y. Xue and D.E. Brown. Spatial analysis with preference specification of latent decision makers for criminal event prediction. *Decision Support Systems*, 41(3):560–573, 2006.
112. Y.Z. Yang and X.S. Gu. Cultural-based genetic tabu algorithm for multiobjective job shop scheduling. *Mathematical Problems in Engineering*, 2014:14, 2014. Article ID 230719.
113. X.H. Zhang, S.Q. Zhong, Y.L. Liu, and X.L. Wang. A framing link based tabu search algorithm for large-scale multidepot vehicle routing problems. *Mathematical Problems in Engineering*, 2014:13, 2014. Article ID 152494.
114. Y. Zhang and D.E. Brown. Police patrol districting method and simulation evaluation using agent-based model & GIS. *Security Informatics*, 2(7), 2013.
115. J.R. Zipkin, M.B. Short, and A.L. Bertozzi. Cops on the dots in a mathematical model of urban crime and police response. 19(5):1479–1506, 2014.
116. N. Zoroa, N.J. Fernandez-Saez, and P. Zoroa. Patrolling a perimeter. *European Journal of Operational Research*, 222(3):571–582, 2012.

Appendices

APPENDIX A

SOURCE CODE

We now present the source code of the solution algorithm for the MC-PCP.

A.1. File: main.cpp

This file contains the *main* method of the program that obtains the inputs (data and parameters) from the users, runs the optimization algorithm and prints the result.

main.cpp

```
#include <iostream>
#include <cstdlib>

#include "InputData.hpp"
#include "MultiStart.hpp"
#include "Parameters.hpp"

using namespace std;

InputData* inputData;
MultiStart* multiStart;

typedef struct {
   int subset_num, cpu, iter;
   char* input_filename;
   char* sol_filename;
   double wDemand, wArea, wDiameter, wSupport, lambda;
} input_t;

void freeMemory(void);

void instructions(char *name){
```

```
    cout << "Usage: " << name << " [OPTION]..." << endl;
    cout << "Partitions a matrix according to efficiency criteria." << endl;
    cout << endl;
    cout << "Mandatory arguments:" << endl;
    cout << "-I,\t -i\t\t input filename." << endl;
    cout << "-p,\t -p\t\t number of partitions [>0]." << endl;
    cout << endl;
    cout << "Optional arguments:" << endl;
    cout << "-WD,\t -wd\t\t weight associated to demand [>=0] (default: " << Parameters::WDEMAND
        << ")." << endl;
    cout << "-WI,\t -wi\t\t weight associated to diameter [>=0] (default: " <<
        Parameters::WDIAMETER << ")." << endl;
    cout << "-WA,\t -wa\t\t weight associated to area [>=0] (default: " << Parameters::WAREA <<
        ")." << endl;
    cout << "-WS,\t -ws\t\t weight associated to support [>=0] (default: " <<
        Parameters::WSUPPORT << ")." << endl;
    cout << "\t\t\t [weights are always normalized]" << endl;
    cout << "-L,\t -l\t\t balance/optimization tradeoff [0-1] (default: " << Parameters::LAMBDA
        << ")." << endl;
    cout << "-CT,\t -ct\t\t optimization CPU time [>0] (default: " << Parameters::CPU << ")." <<
        endl;
    cout << "-IT,\t -it\t\t empty iterations [>0] (default: " << Parameters::ITERATIONS << ")."
        << endl;
    cout << "-S,\t -s\t\t solution filename for its evaluation. When provided the program
        returns" << endl;
    cout << "\t\t\t the evaluation of the solution's attribute." << endl;
    cout << "-H,\t -h\t\t this help." << endl;
    cout << endl;
    cout << "Example:" << endl;
    cout << name << " -i madrid.input -p 6 -L 0.33" << endl;
    cout << "determines 6 subsets over the madrid.input instance, setting lambda to 0.33 and
        using default\nweights." << endl;

    exit(EXIT_SUCCESS);
}

input_t paramConfig(int argc, char *argv[])
{
  input_t input;
  bool p = false, inputfile = false;

  input.wDemand = Parameters::WDEMAND;
  input.wDiameter = Parameters::WDIAMETER;
  input.wArea = Parameters::WAREA;
  input.wSupport = Parameters::WSUPPORT; //DEF_WEIGHT;
  input.lambda = Parameters::LAMBDA; //DEF_LAMBDA;
  input.cpu = Parameters::CPU; //DEF_CPU;
  input.iter = Parameters::ITERATIONS; //DEF_ITER;
  input.sol_filename = NULL;

  int pos = 1;
```

```
while(pos < argc){
   string param = argv[pos];
   if(param == "-P" || param == "-p"){
      input.subset_num = atoi(argv[pos+1]);
      if(input.subset_num <= 0){
         cerr << argv[0] << ": " << param << " accepts only positive values." << endl;
         exit(EXIT_FAILURE);
      }
      p = true;
   }else if(param == "-I" || param == "-i"){
      input.input_filename = argv[pos+1];
      inputfile = true;
   }else if(param == "-IS" || param == "-is"){
      input.sol_filename = argv[pos+1];
   }else if(param == "-WD" || param == "-wd"){
      input.wDemand = atof(argv[pos+1]);
      if(input.wDemand < 0.0f){
         cerr << argv[0] << ": " << param << " accepts only non-negative values." << endl;
         exit(EXIT_FAILURE);
      }
   }else if(param == "-WI" || param == "-wi"){
      input.wDiameter = atof(argv[pos+1]);
      if(input.wDiameter < 0.0f){
         cerr << argv[0] << ": " << param << " accepts only non-negative values." << endl;
         exit(EXIT_FAILURE);
      }
   }else if(param == "-WA" || param == "-wa"){
      input.wArea = atof(argv[pos+1]);
      if(input.wArea < 0.0f){
         cerr << argv[0] << ": " << param << " accepts only non-negative values." << endl;
         exit(EXIT_FAILURE);
      }
   }else if(param == "-WS" || param == "-ws"){
      input.wSupport = atof(argv[pos+1]);
      if(input.wSupport < 0.0f){
         cerr << argv[0] << ": " << param << " accepts only non-negative values." << endl;
         exit(EXIT_FAILURE);
      }
   }else if(param == "-L" || param == "-l"){
      input.lambda = atof(argv[pos+1]);
      if(input.lambda < 0.0f || input.lambda > 1.0f){
         cerr << argv[0] << ": " << param << " accepts only values between 0 and 1." << endl;
         exit(EXIT_FAILURE);
      }
   }else if(param == "-CT" || param == "-ct"){
      input.cpu = atoi(argv[pos+1]);
      if(input.cpu <= 0){
         cerr << argv[0] << ": " << param << " accepts only positive values." << endl;
         exit(EXIT_FAILURE);
      }
   }else if(param == "-IT" || param == "-it"){
```

```
        input.iter = atoi(argv[pos+1]);
        if(input.iter <= 0){
           cerr << argv[0] << ": " << param << " accepts only positive values." << endl;
           exit(EXIT_FAILURE);
        }
     }else if(param == "-S" || param == "-s"){
        input.sol_filename = argv[pos+1];
     }else if(param == "-H" || param == "-h"){
        instructions(argv[0]);
     }else{
        cerr << argv[0] << ": " << param << ": command not found" << endl;
        exit(EXIT_FAILURE);
     }
     pos += 2;
  }

  if(!inputfile) cerr << argv[0] << ": Input filename not specified. See " << argv[0] << " -H."
       << endl;
  if(input.sol_filename == NULL && !p) cerr << argv[0] << ": Number of partitions not specified.
       See " << argv[0] << " -H." << endl;
  if((input.sol_filename == NULL && !p) || !inputfile){
     exit(EXIT_FAILURE);
  }

  //weights normalization
  double sumWeights = input.wDemand + input.wDiameter + input.wArea + input.wSupport;
  input.wDemand /= sumWeights;
  input.wDiameter /= sumWeights;
  input.wArea /= sumWeights;
  input.wSupport /= sumWeights;

  return input;
}

int main(int argc, char *argv[])
{
  input_t input = paramConfig(argc, argv);

  Parameters::WDEMAND = input.wDemand;
  Parameters::WAREA = input.wArea;
  Parameters::WDIAMETER = input.wDiameter;
  Parameters::WSUPPORT = input.wSupport;
  Parameters::LAMBDA = input.lambda;
  Parameters::CPU = input.cpu;
  Parameters::ITERATIONS = input.iter;

  inputData = new InputData(input.subset_num, input.input_filename);
  Parameters::ITERATIONS = inputData->numNodes();
  Parameters::TABU_TENURE = inputData->numNodes();

  if(input.sol_filename == NULL){
```

```
      //Optimization
      multiStart = new MultiStart(inputData);
      multiStart->search();
      freeMemory();
   }else{
      //Solution Analysis
      Partition* sol = new Partition(inputData, input.sol_filename);
      sol->printPartition();
      sol->printResume();
      delete sol;
   }

   exit(EXIT_SUCCESS);
}

//Deallocates all the memory structures created.
void freeMemory(void)
{
   //delete randomSearch;
   delete inputData;
}
```

A.2. Files: Parameters.hpp and Parameters.cpp

The user can use these files to set the default importance weights for the criteria, as well as select the optimization algorithm to use and define the default parameters for the tabu search.

Parameters.hpp

```
#ifndef PARAMETERS_HPP
#define PARAMETERS_HPP

typedef enum improve_e{FIRST_IMPROVE, BEST_IMPROVE} improve_t;
typedef enum search_e{LOCAL_SEARCH, TABU_SEARCH} search_t;

class Parameters{
public:
   //Decision Maker Preferences
   static double WDEMAND;
   static double WAREA;
   static double WDIAMETER;
   static double WSUPPORT;
   static double LAMBDA;

   //Convex unfeasibility penalty
   static double PENALTY;
```

```
    //Search algorithm parameters
    static improve_t IMPROVE_STRATEGY;
    static search_t SEARCH_ALGORITHM;
    static unsigned int TABU_TENURE;

    //Iterative algorithm parameters
    static unsigned int CPU;
    static unsigned int ITERATIONS;
};

#endif
```

Parameters.cpp

```
#include "Parameters.hpp"

double Parameters::WDEMAND = 0.45;
double Parameters::WAREA = 0.45;
double Parameters::WDIAMETER = 0.05;
double Parameters::WSUPPORT = 0.05;
double Parameters::LAMBDA = 0.1;
double Parameters::PENALTY = 1;

improve_t Parameters::IMPROVE_STRATEGY = BEST_IMPROVE; // BEST_IMPROVE or FIRST_IMPROVE
search_t Parameters::SEARCH_ALGORITHM = TABU_SEARCH; // LOCAL_SEARCH or TABU_SEARCH
unsigned int Parameters::TABU_TENURE = 111;
unsigned int Parameters::ITERATIONS = 111;

unsigned int Parameters::CPU = 60;
```

A.3. File: InputData.hpp and InputData.cpp

These files define the data structures used to represent the input data required by the optimization algorithms.

InputData.hpp

```
#ifndef INPUTDATA_HPP
#define INPUTDATA_HPP

#include <string>
#include <vector>

using namespace std;

typedef double crime_t;
```

```
typedef double area_t;
typedef double dist_t;
typedef unsigned int node_t;

/*
 * Class managing input data structures.
 */
class InputData{
private:

   //Number of nodes in the graph
   unsigned int p_nodes;

   //Number of edges in the graph
   unsigned int p_edges;

   //Nodes crime
   crime_t *p_crime;

   //Nodes area
   area_t *p_area;

   //Number of partitions.
   unsigned int p_num_subsets;

   //Acquires the data from an input file.
   //a_InputFile : input file name string.
   void readInputData(string a_InputFile);

   //Total risk
   crime_t p_tot_crime;

   //Total area
   area_t p_tot_area;

   //Nodes adjacency matrix
   bool **p_adj_mat;

   //List of edges by node
   vector<node_t> *p_node_edge;

   //Nodes distance matrix
   dist_t **p_dist_mat;

   //Edge shortest path distance matrix
   unsigned int **p_edgeSP_mat;

   //Shortest path distance matrix
   dist_t **p_distSP_mat;

   //Graph diameter
```

```
    dist_t p_diameter;

    //Maximum support distance
    dist_t p_support_dist;

    //Max edge distance
    dist_t p_max_dist;

    //Big enough constant
    dist_t p_big_M;

public:

    //returns the number of nodes in the graph
    unsigned int numNodes(void);

    //returns the total crime in the graph
    crime_t totalCrime(void);

    //returns the total area in the graph
    area_t totalArea(void);

    //returns the maximum diameter in the graph
    dist_t diameter(void);

    //Data class constructor.
    //a_InputFile : input file name string.
    InputData(unsigned int a_num_partitions, string a_InputFile);

    //Data class destructor.
    ~InputData(void);

    //Returns the crime of a specific node
    crime_t crime(node_t a_node);

    //Returns the area of a specific node
    area_t area(node_t a_node);

    //Returns the number of partitions.
    unsigned int numSubsets(void);

    //Sets the number of partitions.
    void numSubsets(unsigned int p);

    //Returns the edge distance between two nodes
    unsigned int edgeDistance(node_t a_node1, node_t a_node2);

    //Returns the distance between two nodes
    dist_t distance(node_t a_node1, node_t a_node2);

    //Returns the maximum support distance
```

```
   dist_t supportDistance(void);

   //Returns a big enough constant
   dist_t bigM(void);

   //Returns the degree of a node
   unsigned int numEdges(node_t a_node);

   //Returns the destination of the pos-th edge exiting from a_node
   node_t edgeByPos(node_t a_node, unsigned int pos);

   //Returns true if there exists an edge connecting the two nodes
   bool isEdge(node_t a_nodeA, node_t a_nodeB);
};

#endif
```

InputData.cpp

```
#include <cstdlib>
#include <iostream>
#include <fstream>
#include <cmath>

#include "InputData.hpp"
#include "mymemory.hpp"
#include "myalgorithm.hpp"

InputData::InputData(unsigned int a_num_subsets, string a_InputFile)
{
   p_num_subsets = a_num_subsets;

   //Reading input file
   readInputData(a_InputFile);

   //Building data structures
   //Edge shortest path distance matrix
   //and shortest path distance matrix
   p_edgeSP_mat = new_matrix<unsigned int>(p_nodes, p_nodes);
   p_distSP_mat = new_matrix<dist_t>(p_nodes, p_nodes);
   for(node_t i = 0; i < p_nodes; ++i){
      for(node_t j = 0; j < p_nodes; ++j){
         if(i == j){
            p_edgeSP_mat[i][j] = p_edgeSP_mat[j][i] = 0;
         }else{
            p_edgeSP_mat[i][j] = p_adj_mat[i][j]?1:(unsigned int)p_big_M;
            p_distSP_mat[i][j] = p_adj_mat[i][j]?p_dist_mat[i][j]:p_big_M;
         }
      }
   }
```

```
   FloydWarshall<unsigned int>(p_edgeSP_mat, p_nodes);
   FloydWarshall<dist_t>(p_distSP_mat, p_nodes);

   //Computing parameters
   //Graph diameter
   p_diameter = 0;
   for(node_t i = 0; i < p_nodes-1; ++i){
      for(node_t j = i+1; j < p_nodes; ++j){
         if(p_distSP_mat[i][j] > p_diameter)
            p_diameter = p_distSP_mat[i][j];
      }
   }

   //Maximum support distance
   p_support_dist = p_diameter/ (2*sqrt((dist_t)p_num_subsets));
}

InputData::~InputData(void)
{
   if(p_crime != NULL) delete[] p_crime;
   if(p_area != NULL) delete[] p_area;
   if(p_node_edge != NULL) delete[] p_node_edge;

   delete_matrix(p_adj_mat, p_nodes);
   delete_matrix(p_dist_mat, p_nodes);
   delete_matrix(p_edgeSP_mat, p_nodes);
   delete_matrix(p_distSP_mat, p_nodes);
}

void InputData::readInputData(string a_InputFile)
{
   //Open the input file.
   ifstream in_file_stream;
   in_file_stream.open(a_InputFile.c_str());

   if(!in_file_stream.is_open())
   {
      cerr << "ERROR in InputData::readInputData : input file cannot be opened." << endl;
      exit(EXIT_FAILURE);
   }

   //Read the number of nodes
   in_file_stream >> p_nodes;

   //Instantiate crime array
   p_crime = new crime_t[p_nodes];
   p_tot_crime = 0;

   //Read crime data from file
   for(node_t i = 0; i < p_nodes; ++i)
   {
```

```
   in_file_stream >> p_crime[i];
   p_tot_crime += p_crime[i];
}

//Instantiate area array
p_area = new area_t[p_nodes];
p_tot_area = 0;

//Read area data from file
for(node_t i = 0; i < p_nodes; ++i)
{
   in_file_stream >> p_area[i];
   p_tot_area += p_area[i];
}

//Read the number of edges
in_file_stream >> p_edges;

//Instantiate list of edges by node
p_node_edge = new vector<node_t>[p_nodes];

//Instantiate adjacency matrix
p_adj_mat = new_matrix<bool>(p_nodes, p_nodes);

//Instantiate distance matrix
p_dist_mat = new_matrix<dist_t>(p_nodes, p_nodes);

for(unsigned int i = 0; i < p_nodes; ++i){
   for(unsigned int j = 0; j < p_nodes; ++j){
      p_adj_mat[i][j] = false;
      p_dist_mat[i][j] = 0;
   }
}

//Reading edges
node_t edgeA, edgeB;
dist_t dist;
p_max_dist = 0;
for(unsigned int i = 0; i < p_edges; ++i){
   in_file_stream >> edgeA >> edgeB >> dist;
   --edgeA; --edgeB;
   if(dist > p_max_dist) p_max_dist = dist;
   p_adj_mat[edgeA][edgeB] = p_adj_mat[edgeB][edgeA] = true;
   p_dist_mat[edgeA][edgeB] = p_dist_mat[edgeB][edgeA] = dist;
   p_node_edge[edgeA].push_back(edgeB);
   p_node_edge[edgeB].push_back(edgeA);
}

p_big_M = p_max_dist * p_edges + 1;

in_file_stream.close();
```

```
}

crime_t InputData::crime(node_t a_node)
{
   if(a_node < p_nodes){
      return p_crime[a_node];
   }else{
      cerr << "ERROR in InputData::getCrime : argument our of range." << endl;
      exit(EXIT_FAILURE);
   }
}

area_t InputData::area(node_t a_node)
{
   if(a_node < p_nodes){
      return p_area[a_node];
   }else{
      cerr << "ERROR in InputData::getArea : argument out of range." << endl;
      exit(EXIT_FAILURE);
   }
}

unsigned int InputData::edgeDistance(node_t a_node1, node_t a_node2){
   if(a_node1 < p_nodes && a_node2 < p_nodes){
      return p_edgeSP_mat[a_node1][a_node2];
   }else{
      cerr << "ERROR in InputData::edgeDistance : argument out of range." << endl;
      exit(EXIT_FAILURE);
   }
}

dist_t InputData::distance(node_t a_node1, node_t a_node2){
   if(a_node1 < p_nodes && a_node2 < p_nodes){
      return p_distSP_mat[a_node1][a_node2];
   }else{
      cerr << "ERROR in InputData::distance : argument out of range." << endl;
      exit(EXIT_FAILURE);
   }
}

unsigned int InputData::numNodes(void){return p_nodes;}

unsigned int InputData::numSubsets(void){return p_num_subsets;}

void InputData::numSubsets(unsigned int p){p_num_subsets = p;}

crime_t InputData::totalCrime(void){return p_tot_crime;}

area_t InputData::totalArea(void){return p_tot_area;}

dist_t InputData::diameter(void){return p_diameter;}
```

```
dist_t InputData::supportDistance(void){return p_support_dist;}

dist_t InputData::bigM(void){return p_big_M;}

unsigned int InputData::numEdges(node_t a_node){
   return p_node_edge[a_node].size();
}

node_t InputData::edgeByPos(node_t a_node, unsigned int pos){
   return p_node_edge[a_node][pos];
}

bool InputData::isEdge(node_t a_nodeA, node_t a_nodeB){
   return p_adj_mat[a_nodeA][a_nodeB];
}
```

A.4. File: MultiStart.hpp and MultiStart.cpp

MultiStart.cpp provides the framework for the implementation of the random multi-start optimization algorithm. It iteratively calls the greedy constructive heuristics and the chosen local search optimization algorithm, until the stopping condition is met.

MultiStart.hpp

```
#ifndef MULTISTART_CPP
#define MULTISTART_CPP

#include "Partition.hpp"
#include "InputData.hpp"

class MultiStart
{
private:
   InputData* p_input_data;
   Partition* p_best_partition;

public:
   MultiStart(InputData* a_input_data);
   ~MultiStart(void);
   void search(void);
   Partition* bestPartition(void);
};
#endif
```

MultiStart.cpp

```
#include <cmath>
#include <cstdlib>
#include <iostream>
#include <ctime>

#include "MultiStart.hpp"
#include "Parameters.hpp"
#include "SearchAlgorithm.hpp"

using namespace std;

MultiStart::MultiStart(InputData* a_input_data)
{
   srand ((unsigned int)time(NULL));

   p_input_data = a_input_data;
   p_best_partition = NULL;
}

MultiStart::~MultiStart(void)
{
   if(p_best_partition != NULL)
      delete p_best_partition;
}

void MultiStart::search(void)
{
   Partition* sol;
   unsigned int emptyIter = 0;

   time_t timeStart = time(NULL);

   SearchAlgorithm optAlgorithm(p_input_data);

   //Loop until the time is up.
   while(time(NULL) - timeStart < Parameters::CPU)// &&
      // (p_best_partition == NULL || emptyIter < Parameters::ITERATIONS))
   {

      //Create the new solution using
      //a constructive greedy heuristic
      sol = optAlgorithm.constrHeuristic();

      if(!sol->isValid() ||
         (p_best_partition != NULL && sol->value() > p_best_partition->value() * 1.50))
      {
         delete sol;
         sol = NULL;
         //++emptyIter;
         continue;
```

```
    }

    //Local search optimization
    optAlgorithm.optimize(sol, Parameters::CPU - time(NULL) + timeStart);
    //cout << "NEW SOLUTION FOUND (" << time(NULL) - timeStart << "s) : "<< sol->value() << endl;
    if(p_best_partition == NULL || sol->isBetterThan(*p_best_partition))
    {
       if(p_best_partition != NULL)
          delete p_best_partition;
       p_best_partition = sol;
       //cout << "NEW SOLUTION FOUND (" << time(NULL) - timeStart << "s) : "<<
            p_best_partition->value() << endl;
       p_best_partition->timeStamp = time(NULL) - timeStart;
       //p_best_partition->printPartition();
       emptyIter = 0;
    }
    else{
       delete sol;
       ++emptyIter;
    }
    sol = NULL;
  }
  //cout << "TOTAL TIME : " << time(NULL) - timeStart << "s\t" << "EMPTY ITERs : " << emptyIter
       << endl;
  p_best_partition->printConfiguration();
  p_best_partition->printResume();
}

Partition* MultiStart::bestPartition(void)
{
  return p_best_partition;
}
```

A.5. File: SearchAlgorithm.hpp and SearchAlgorithm.cpp

The greedy optimization algorithm and the local search procedures are described in these files. Also, the data structure used to represent the tabu tenure is provided here.

SearchAlgorithm.hpp

```
#if !defined SEARCHALGORITHM_HPP
#define SEARCHALGORITHM_HPP

#include <vector>
#include <cstring>
#include "InputData.hpp"
```

```
#include "Partition.hpp"

//Class that defines a standard move in the search algorithm
class Move{
public:
   unsigned int from_p, to_p;
   node_t node;
   Move(unsigned int a_from, unsigned int a_to, node_t a_node):
   from_p(a_from), to_p(a_to), node(a_node){};
};

class TabuTenureElem{
public:
   int *solution;
   unsigned int size;
   unsigned int expiration;

   bool isSame(int *a_solution){
      bool isSame = true;
      for(unsigned int i = 0; isSame && i < size; ++i){
         if(solution[i] != a_solution[i]){
            isSame = false;
         }
      }

      return isSame;
   }

   TabuTenureElem(int* a_solution, unsigned int a_size, unsigned int a_expiration){
      expiration = a_expiration;
      size = a_size;
      solution = new int[size];
      memcpy(solution, a_solution, sizeof(int) * size);
   }

   ~TabuTenureElem(){
      if(solution != NULL)
         delete[] solution;
   }
};

class SearchAlgorithm{
private:
   InputData *p_input_data;
   vector<TabuTenureElem*> p_tabuTenure;

   void buildMoveList(Partition *a_solution, vector<Move> &a_moveList);

   //Check if the solution is tabu
   bool isTabu(Partition *a_solution);
```

```
   //Decrease the expiration counter of the solutions in the tabu tenure
   void TenureExpire(void);

   //Clear the Tabu Tenure
   void EmptyTenure(void);

   //Add a solution to the tabu tenure
   void AddToTenure(Partition *a_solution);

public:

   SearchAlgorithm(InputData *a_inputData);
   ~SearchAlgorithm(void);

   //Creates a new partition by constructive greedy heuristic.
   Partition* constrHeuristic(void);

   //Optimizes a partition using local search.
   void optimize(Partition* a_solution, time_t a_time);

};

#endif
```

SearchAlgorithm.cpp

```
#include <vector>
#include <algorithm>
#include <iostream>
#include <ctime>
#include "Parameters.hpp"
#include "SearchAlgorithm.hpp"

SearchAlgorithm::SearchAlgorithm(InputData *a_inputData){
   p_input_data = a_inputData;
}

SearchAlgorithm::~SearchAlgorithm(void){
   if(Parameters::SEARCH_ALGORITHM == TABU_SEARCH)
      EmptyTenure();
}

void SearchAlgorithm::buildMoveList(Partition *a_solution, vector<Move> &a_moveList){
   node_t node;
   unsigned int pFrom;

   a_moveList.clear();
   for(unsigned int pTo = 0; pTo < p_input_data->numSubsets(); ++pTo)
   {
      for(unsigned int pos = 0; pos < a_solution->p_partition[pTo]->numNeighElements(); ++pos){
```

```
        node = a_solution->p_partition[pTo]->getNeighbor(pos);
        pFrom = a_solution->p_solution[node];
        if(a_solution->p_partition[pFrom]->numElements() > 1){
           a_moveList.push_back(Move(pFrom,pTo,node));
        }
     }
   }
   random_shuffle(a_moveList.begin(), a_moveList.end());
}

bool SearchAlgorithm::isTabu(Partition *a_solution){
   bool found = false;
   //Search the current solution
   for(unsigned int i = 0; !found && i < p_tabuTenure.size(); ++i){
      found = p_tabuTenure[i]->isSame(a_solution->p_solution);
      if(found) p_tabuTenure[i]->expiration = Parameters::TABU_TENURE;
   }
   return found;
}

void SearchAlgorithm::EmptyTenure(void){
   for(unsigned int i = 0; i < p_tabuTenure.size(); ++i){
      delete p_tabuTenure[i];
   }
   p_tabuTenure.clear();
}

void SearchAlgorithm::AddToTenure(Partition *a_solution){
   //We assume that the solution is not already in the tabu tenure
   TabuTenureElem *elem = new TabuTenureElem(a_solution->p_solution, p_input_data->numNodes(),
        Parameters::TABU_TENURE);
   p_tabuTenure.push_back(elem);
}

void SearchAlgorithm::TenureExpire(void){
   unsigned int i = 0;

   //Decrease the expiration of all the solutions in the tabu tenure
   while(i < p_tabuTenure.size()){
      p_tabuTenure[i]->expiration--;
      if(p_tabuTenure[i]->expiration == 0){
         //Remove the solutions that have expired
         p_tabuTenure.erase(p_tabuTenure.begin()+i);
      }else ++i;
   }
}

void SearchAlgorithm::optimize(Partition *a_solution, time_t a_time)
{
   bool loopCondition, improved;
   Partition *current_sol, *new_sol, *bestLocal_sol, *bestGlobal_sol;
```

```
vector<Move> moveList;

current_sol = new Partition(*a_solution);
bestGlobal_sol = current_sol;
new_sol = bestLocal_sol = NULL;
loopCondition = true;

time_t timeStart = time(NULL);

unsigned int emptyIter = 0;
//Loop while the conditions is met.
while(loopCondition &&
   time(NULL) - timeStart < a_time &&
   emptyIter < Parameters::ITERATIONS)
{
   loopCondition = false;

   //Update the expiration time of tabu tenure
   if(Parameters::SEARCH_ALGORITHM == TABU_SEARCH){
      TenureExpire();
      //Add current solution to tabu tenure
      AddToTenure(current_sol);
   }

   //Create the vector of all the possible exchanges
   buildMoveList(current_sol, moveList);

   improved = false;
   bestLocal_sol = NULL;
   //Loop on all the elements belonging to any subset, in random order
   while((Parameters::IMPROVE_STRATEGY == BEST_IMPROVE && !moveList.empty()) ||
      (Parameters::IMPROVE_STRATEGY == FIRST_IMPROVE && !improved && !moveList.empty()) )
   {
      Move tmp_move = moveList.back();
      moveList.pop_back();

      //Generate the neighbor solution
      //Copy the current solution
      new_sol = new Partition(*current_sol);
      //Execute the move
      new_sol->p_partition[tmp_move.from_p]->remove(tmp_move.node);
      new_sol->p_partition[tmp_move.to_p]->add(tmp_move.node, false);
      new_sol->p_solution[tmp_move.node] = tmp_move.to_p;
      //Normalize solution
      new_sol->normalize();
      //Evaluate the fitness of the new solution
      new_sol->computeValue();

      //Check if the solution can be considered
      if(Parameters::SEARCH_ALGORITHM == LOCAL_SEARCH ||
         (Parameters::SEARCH_ALGORITHM == TABU_SEARCH && !isTabu(new_sol)))
```

```
      {
         //Update the best local solution if necessary
         if(bestLocal_sol == NULL ||
            new_sol->isBetterThan(*bestLocal_sol)){
            delete(bestLocal_sol);
            bestLocal_sol = new_sol;

            //Update the best global solution if necessary
            if(bestLocal_sol->isBetterThan(*bestGlobal_sol))
               improved = true;
         }
      }

      //Check if the new solution can be destroyed
      if(new_sol != bestLocal_sol)
         delete new_sol;
      new_sol = NULL;
   }

   //Update best global solution with best local if necessary
   if(bestLocal_sol != NULL &&
      (improved ||
      Parameters::SEARCH_ALGORITHM == TABU_SEARCH)){

      if(current_sol != bestGlobal_sol)
         delete current_sol;
      current_sol = bestLocal_sol;

      if(bestLocal_sol->isBetterThan(*bestGlobal_sol)){
         delete bestGlobal_sol;
         bestGlobal_sol = bestLocal_sol;
      }
      bestLocal_sol = NULL;
      loopCondition = true;
   }else{
      if(bestLocal_sol != bestGlobal_sol) delete bestLocal_sol;
      if(current_sol != bestGlobal_sol) delete current_sol;
      bestLocal_sol = current_sol = NULL;
   }

   if(improved) emptyIter = 0;
   else ++emptyIter;
}//End while(LoopCondition)

//Update the solution with the best solution found
if(bestGlobal_sol->isBetterThan(*a_solution)){
   a_solution->copyPartition(*bestGlobal_sol);
}
if(current_sol != bestGlobal_sol){
   if(bestLocal_sol != current_sol) delete bestLocal_sol;
   delete current_sol;
```

```
   }else if(bestLocal_sol != bestGlobal_sol) delete bestLocal_sol;

   delete bestGlobal_sol;
   //cout << "\n" << endl;
}


Partition* SearchAlgorithm::constrHeuristic(void)
{
   Partition *solution = new Partition(p_input_data);

   //STEP ONE
   //Assign a random node to each subset.
   vector<node_t> nodeList;
   for(node_t i = 0; i < p_input_data->numNodes(); ++i){
      nodeList.push_back(i);
   }

   unsigned int rnd;
   node_t node;
   for(unsigned int sub1 = 0; sub1 < p_input_data->numSubsets(); ++sub1)
   {
      rnd = rand() % (p_input_data->numNodes() - sub1);
      node = nodeList[rnd];
      nodeList.erase(nodeList.begin()+rnd);

      //Add the random node to the subset
      solution->p_partition[sub1]->add(node, false);
      solution->p_solution[node] = sub1;
   }
   nodeList.clear();

   //STEP TWO
   //Expand the subsets in a greedy fashion.
   unsigned int count = p_input_data->numSubsets();
   Partition *tmp_partition = NULL, *best_partition = NULL;
   vector<unsigned int> subsetList;
   unsigned int set;

   //Loop on all the remaining nodes
   while(count < p_input_data->numNodes())
   {
      //Include all the non-assigned neighboring nodes into the corresponding subset
      //and select the best one.
      for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub){
         for(unsigned int pos = 0; pos < solution->p_partition[sub]->numNeighElements(); ++pos){
            if(solution->p_solution[solution->p_partition[sub]->getNeighbor(pos)] == -1){
               nodeList.push_back(solution->p_partition[sub]->getNeighbor(pos));
               subsetList.push_back(sub);
            }
         }
```

```
    }

    //Loop on all the neighbor nodes, randomly chosen
    while(!subsetList.empty())
    {
       rnd = rand()%subsetList.size();
       set = subsetList[rnd];
       subsetList.erase(subsetList.begin()+rnd);
       node = nodeList[rnd];
       nodeList.erase(nodeList.begin()+rnd);

       //Copy the current partition
       tmp_partition = new Partition(*solution);
       //Include the node to the current subset
       tmp_partition->p_partition[set]->add(node, false);
       tmp_partition->p_solution[node] = set;
       //TODO:if(tmp_partition->isConvex(set))
       {
          //Evaluate the fitness of the new partition
          tmp_partition->computeValue();
          //Update the best partition found
          if(best_partition == NULL ||
             tmp_partition->isBetterThan(*best_partition))
          {
             if(best_partition != NULL)
                delete best_partition;
             best_partition = tmp_partition;
          }
       }
       if(tmp_partition != best_partition)
          delete tmp_partition;
       tmp_partition = NULL;
    }

    //Update current partition with the best one.
    if(best_partition != NULL)
    {
       solution->copyPartition(*best_partition);
       delete best_partition;
       best_partition = NULL;
    }
    else
    {
       solution->p_valid = false;
       return NULL;
    }

    //Increase the counter
    ++count;

  }
```

```
   solution->p_valid = true;
   solution->normalize();

   return solution;
}
```

A.6. File: Partition.hpp and Partition.cpp

These file describe the class *Partition* that provides methods and fields for the representation of a partition.

Partition.hpp

```
#ifndef PARTITION_HPP
#define PARTITION_HPP

#include "InputData.hpp"
#include "Subset.hpp"


/*
 * Class representing a valid partition of the demand matrix.
 */
class Partition
{
   friend class SearchAlgorithm;

private:
   //Input data.
   InputData* p_input_data;

   //The partition represented as an array of subsets.
   Subset** p_partition;

   //True if the partition is a valid one. False otherwise.
   bool p_valid;

   //Partition objective function value.
   double p_value;

   //List of subsets by node
   int *p_solution;

   //Calculates the fitness of the current partition.
   void calculateFitness(void);
```

```
  //Copies a partition into the current one.
  void copyPartition(Partition& a_other);

  //Update the attributes values.
  void computeValue(void);

  //Returns true if all the subsets are convex.
  //False otherwise.
  bool isConvex(void);

  //Returns true if subsets a_sub is convex.
  bool isConvex(unsigned int a_sub);

  //Normalize the solution to avoid simmetries
  void normalize(void);

public:

  //Time stamp representing when the solution was generated
  time_t timeStamp;

  //Constructor of the class.
  //a_input_data : pointer to the input data.
  Partition(InputData *a_input_data);

  //Constructor of the class.
  //a_input_data : pointer to the input data.
  //a_sol_filename : name of the file storing the partition.
  Partition(InputData *a_input_data, string a_sol_filename);

  //Copy constructor of the class.
  //a_other : original element to be copied.
  Partition(Partition& a_other);

  //Destructor of the class.
  ~Partition(void);

  //Prints the partition and attribute values to the out stream.
  void printPartition(void);

  //Prints exclusively the partition to the err stream.
  void printConfiguration(void);

  //Returns true if the current partition is better than or equal to the other.
  //Returns false otherwise.
  bool isBetterThan(const Partition& a_other) const;

  //Returns true if the partition has been properly generated.
  //False otherwise.
  bool isValid(void);
```

```
   //Prints a resume of the partition stats
   void printResume(void);

   //Returns the value of the partition
   double value(void);
};
#endif
```

Partition.cpp

```
#include <cstdlib>
#include <cmath>
#include <iostream>
#include <fstream>
#include <limits>
#include <cstring>
#include "Partition.hpp"
#include "Parameters.hpp"

using namespace std;

Partition::Partition(InputData *a_input_data)
{
   p_input_data = a_input_data;
   p_partition = new Subset*[p_input_data->numSubsets()];
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
   {
      p_partition[sub] = new Subset(a_input_data);
      p_solution = NULL;
   }
   p_solution = new int[p_input_data->numNodes()];
   for(node_t node = 0; node < p_input_data->numNodes(); ++node){
      p_solution[node] = -1;
   }

   p_value = 0;
   p_valid = false;
}

Partition::Partition(InputData *a_input_data, string a_sol_filename)
{
   p_input_data = a_input_data;

   //Open the partition file.
   ifstream sol_file_stream;
   sol_file_stream.open(a_sol_filename.c_str());

   if(!sol_file_stream.is_open())
   {
      cerr << "ERROR in Partition::Partition : solution file cannot be opened." << endl;
```

```
    exit(EXIT_FAILURE);
  }

  int num;
  int p = 0;
  for(unsigned int i = 0; i < a_input_data->numNodes(); ++i){
    sol_file_stream >> num;
    if(num > p) p = num;
  }
  sol_file_stream.close();

  if(p <= 0)
  {
    cerr << "ERROR in Partition::Partition : solution file not correct. Negative number of
        partitions." << endl;
  }
  p_input_data->numSubsets(p);

  p_partition = new Subset*[p_input_data->numSubsets()];
  for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
  {
    p_partition[sub] = new Subset(a_input_data);
  }
  p_value = 0;
  p_valid = false;

  //Open the partition file.
  sol_file_stream.open(a_sol_filename.c_str());

  if(!sol_file_stream.is_open())
  {
    std::cerr << "ERROR in Partition::Partition : solution file cannot be opened." << endl;
    exit(EXIT_FAILURE);
  }

  p_solution = new int[p_input_data->numNodes()];

  //Read partition list from file
  int par;
  for(unsigned int i = 0; i < a_input_data->numNodes(); ++i)
  {
    sol_file_stream >> par; par--;

    //Checking for errors
    if(par < 0){
      std::cerr << "ERROR in Partition::Partition : solution file element " << i<< " not
          correct." << endl;
      exit(EXIT_FAILURE);
    }

    if(par >= 0)
```

```
      {
         p_partition[par]->add(i, true);
         p_solution[i] = i;
      }
   }

   sol_file_stream.close();

   this->computeValue();
}

Partition::Partition(Partition& a_other)
{
   p_input_data = a_other.p_input_data;
   p_partition = new Subset*[p_input_data->numSubsets()];
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
      p_partition[sub] = NULL;
   p_solution = new int[p_input_data->numNodes()];
   copyPartition(a_other);
   /*
   p_input_data = a_other.p_input_data;
   p_partition = new Subset*[p_input_data->numSubsets()];
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
      p_partition[sub] = new Subset(*(a_other.p_partition[sub]));

   p_solution = new int[p_input_data->numNodes()];
   for(node_t node = 0; node < p_input_data->numNodes(); ++node){
      p_solution[node] = a_other.p_solution[node];
   }

   p_value = a_other.p_value;
   p_valid = false;
   */
}

void Partition::copyPartition(Partition& a_other)
{
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
      if(p_partition[sub] != NULL)
         delete p_partition[sub];

   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
      p_partition[sub] = new Subset(*(a_other.p_partition[sub]));

   memcpy((void*)p_solution, (const void*)a_other.p_solution,
        p_input_data->numNodes()*sizeof(int));

   p_value = a_other.p_value;
   p_valid = a_other.p_valid;

   /*
```

```
   p_input_data = a_other.p_input_data;

   if(p_partition != NULL)
      for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
         delete p_partition[sub];
   delete[] p_partition;

   p_partition = new Subset*[p_input_data->numSubsets()];
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
      p_partition[sub] = new Subset(*(a_other.p_partition[sub]));

   for(node_t node = 0; node < p_input_data->numNodes(); ++node){
      p_solution[node] = a_other.p_solution[node];
   }

   p_value = a_other.p_value;
   p_valid = a_other.p_valid;

   */
}

Partition::~Partition(void)
{
   if(p_partition != NULL)
   {
      for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
         delete p_partition[sub];
      delete[] p_partition;
   }
   if(p_solution != NULL) delete[] p_solution;
}

bool Partition::isConvex(void)
{
   for(unsigned int i = 0; i < p_input_data->numSubsets(); ++i)
      if(!p_partition[i]->isConvex())
         return false;

   return true;
}

void Partition::normalize(void){
   unsigned int sub = 0;
   for(unsigned int i = 0; sub < p_input_data->numSubsets() && i < p_input_data->numNodes(); ++i){
      if(p_solution[i] <(int) sub){
         continue;
      }
      if(p_solution[i] == sub){
         ++sub;
         continue;
      }
```

```
      if(p_solution[i] >(int) sub){
         unsigned int old_sub = p_solution[i];
         Subset *tmp_sub = p_partition[old_sub];
         p_partition[old_sub] = p_partition[sub];
         p_partition[sub] = tmp_sub;
         for(unsigned int j = i; j < p_input_data->numNodes(); ++j){
            if(p_solution[j] == old_sub) p_solution[j] = sub;
            else if(p_solution[j] == sub) p_solution[j] = old_sub;
         }
         ++sub;
      }

   }
}

bool Partition::isConvex(unsigned int a_sub)
{
   if(a_sub >= p_input_data->numSubsets()){
      std::cerr << "ERROR in Partition::isConvex : subset number is not valid." << endl;
      exit(EXIT_FAILURE);
   }
   if(!p_partition[a_sub]->isConvex())
      return false;

   return true;
}

double Partition::value(void){return p_value;}

void Partition::computeValue(void)
{
   //Calculate the number of supporting areas for all the subsets
   unsigned int supporting;
   for(unsigned int sub1 = 0; sub1 < p_input_data->numSubsets(); ++sub1)
   {
      supporting = 0;
      for(unsigned int sub2 = 0; sub2 < p_input_data->numSubsets(); ++sub2)
      {
         if(sub1 == sub2) continue;
         if(p_input_data->distance(p_partition[sub1]->centerMass(),
              p_partition[sub2]->centerMass()) <= p_input_data->supportDistance())
            ++supporting;
      }
      p_partition[sub1]->setSupport(supporting);
   }

   wload_t tmpWLoad, sumWLoad = 0;
   wload_t maxWLoad = - numeric_limits<wload_t>::max();
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
   {
      tmpWLoad = p_partition[sub]->calculateWLoad();
```

```
      sumWLoad += tmpWLoad;
      if(tmpWLoad > maxWLoad) maxWLoad = tmpWLoad;
   }
   p_value = Parameters::LAMBDA * maxWLoad + (1 - Parameters::LAMBDA) * sumWLoad /
        p_input_data->numSubsets();

   //Computing convexity violation
   for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
   {
      if(!p_partition[sub]->isConvex()){
         p_value += Parameters::PENALTY;
      }
   }
}

bool Partition::isBetterThan(const Partition& a_other) const
{
   if(p_value < a_other.p_value)
      return true;

   return false;
}

void Partition::printPartition(void)
{
   printConfiguration();

   cout << "\tAREA\tSUPPORT\tDEMAND\tDIAMET\tLAMBDA\tSUP_DIST" << endl;
   cout << "WEIGHTS" << "\t" << Parameters::WAREA << "\t" << Parameters::WSUPPORT << "\t" <<
        Parameters::WDEMAND << "\t" << Parameters::WDIAMETER << "\t"<< Parameters::LAMBDA << "\t"
        << "(dist:"<< p_input_data->supportDistance()<< ")" << endl;
   for(unsigned int s = 0; s < p_input_data->numSubsets(); ++s)
      cout << s+1 << "\t" << p_partition[s]->area() << "\t" << p_partition[s]->support() << "\t"
           << p_partition[s]->crime() << "\t" << p_partition[s]->diameter() << endl;
   cout << "\t" << endl;
}

void Partition::printConfiguration(void)
{
   for(node_t node = 0; node < p_input_data->numNodes(); ++node ){
      for(unsigned int sub = 0; sub < p_input_data->numSubsets(); ++sub)
      {
         if(p_partition[sub]->isContain(node))
         {
            std::cout << " " << sub+1;
            break;
         }
      }

   }
   cout << endl;
```

```
}

void Partition::printResume(void)
{

   double maxDemand = 0.0f, avgDemand = 0.0f;
   double maxArea = 0.0f, avgArea = 0.0f;
   double maxDiameter = 0.0f, avgDiameter = 0.0f;
   double minSupport = p_input_data->numSubsets()-1, avgSupport = 0.0f;

   for(unsigned int s = 0; s < p_input_data->numSubsets(); ++s){
      avgDemand += p_partition[s]->crime();
      avgArea += p_partition[s]->area();
      avgDiameter += p_partition[s]->diameter();
      avgSupport += p_partition[s]->support();
      if(p_partition[s]->crime() > maxDemand) maxDemand = p_partition[s]->crime();
      if(p_partition[s]->area() > maxArea) maxArea = p_partition[s]->area();
      if(p_partition[s]->diameter() > maxDiameter) maxDiameter = p_partition[s]->diameter();
      if(p_partition[s]->support() < minSupport) minSupport = p_partition[s]->support();
   }

   avgDemand /= p_input_data->numSubsets();
   avgArea /= p_input_data->numSubsets();
   avgDiameter /= p_input_data->numSubsets();
   avgSupport /= p_input_data->numSubsets();

   cerr << p_input_data->numSubsets() << "\t" << Parameters::WAREA << "\t" <<
        Parameters::WSUPPORT << "\t" << Parameters::WDEMAND << "\t" << Parameters::WDIAMETER <<
        "\t" << Parameters::LAMBDA << "\t" << avgArea << "\t" << maxArea << "\t" << avgSupport <<
        "\t" << minSupport << "\t" << avgDemand << "\t" << maxDemand << "\t" << avgDiameter <<
        "\t" << maxDiameter << "\t" << p_value << "\t" << timeStamp << endl;
}

bool Partition::isValid(void)
{
   return p_valid;
}
```

A.7. File: Subset.hpp and Subset.cpp

These files define the class *Subset*. The class *Subset* describes a single patrol sector and provides methods for its evaluation as well all the data required by the optimization algorithm.

Subset.hpp

```
#ifndef SUBSET_HPP
```

```
#define SUBSET_HPP

#include <vector>
#include "InputData.hpp"

using namespace std;

typedef double wload_t;

/*
 * A subset belonging to a partition.
 */
class Subset
{
private:
   //Input data.
   InputData* p_input_data;
   //Nodes belonging to the subset
   vector<node_t> p_node_list;

   //ATTRIBUTES
   //Total demand
   crime_t p_crime;
   //Area
   area_t p_area;
   //Diameter
   dist_t p_diameter;
   //Number of supporting areas
   unsigned int p_support;

   //SUPPORT DATA STRUCTURES AND METHODS
   //Nodes belonging to the neighborhood
   vector<node_t> p_neighbor_list;
   //Updates the neighbor cell list.
   void updateNeighborList(node_t a_node, bool force);
   void computeNeighborList(void);
   //Distance matrix
   dist_t** p_node_dist;
   unsigned int** p_node_edgeDist;
   unsigned int p_node_dist_size;
   //Updates the distance matrix
   void updateDistanceMatrix(void);
   void updateDiameter(void);
   //Center of mass
   node_t p_center_mass;
   void updateCenterMass(void);


public:
   //Constructor
   Subset(InputData *a_input_data);
```

```
Subset(Subset& a_other);

//Destructor
~Subset(void);

//Check if an element belongs to the subset.
bool isContain(node_t a_node);

//Check if an element is a neighbor of the subset.
bool isNeighbor(node_t a_node);

//Check if the subset is connected.
bool isConnected(void);

//Check if the subset is convex.
bool isConvex(void);

//Add an element to the subset.
void add(node_t a_node, bool force);

//Remove an element from the subset.
void remove(node_t a_node);

//Returns the number of elements in the subset.
unsigned int numElements(void);

//Returns an element at a specific position.
node_t get(unsigned int a_pos);

//Returns the number of elements in the neighbor list.
unsigned int numNeighElements(void);

//Returns an element of the neighbor list at a specific position.
node_t getNeighbor(unsigned int a_pos);

//Returns the total demand.
crime_t crime(void);

//Returns the total area.
area_t area(void);

//Returns the diameter.
dist_t diameter(void);

//Returns the shortest patrol route length
dist_t patrolRouteLength(void);

//Returns the number of supporting areas
unsigned int support(void);

//Sets the number of supporting areas
```

```
   void setSupport(unsigned int a_support);

   //Returns the center of mass
   node_t centerMass(void);

   //Computes the total load of the subset, according to the attributes and the weights
   wload_t calculateWLoad(void);

};

#endif
```

Subset.cpp

```
#include <cmath>
#include <iostream>
#include <cstdlib>
#include <limits>

#include "Subset.hpp"
#include "Parameters.hpp"
#include "mymemory.hpp"
#include "myalgorithm.hpp"

#define EPS 1E-4

Subset::Subset(InputData *a_input_data)
{
   p_input_data = a_input_data;
   p_crime = 0;
   p_area = 0;
   p_node_dist = NULL;
   p_node_edgeDist = NULL;
   p_node_dist_size = 0;
   p_diameter = 0;
   p_center_mass = 0;
   p_support = 0;
}

Subset::Subset(Subset& a_other)
{
   p_input_data = a_other.p_input_data;

   //Copy nodes list
   p_node_list = a_other.p_node_list;

   //Copy neighbor list
   p_neighbor_list = a_other.p_neighbor_list;

   //Copy attributes
```

```
   p_crime = a_other.p_crime;
   p_diameter = a_other.p_diameter;
   p_area = a_other.p_area;
   p_center_mass = a_other.p_center_mass;
   p_support = a_other.p_support;

   //Copy distance matrix
   p_node_dist = copy_matrix<dist_t>(a_other.p_node_dist, p_node_list.size(),p_node_list.size());
   p_node_edgeDist = copy_matrix<unsigned int>(a_other.p_node_edgeDist,
        p_node_list.size(),p_node_list.size());

   p_node_dist_size = a_other.p_node_dist_size;
}

Subset::~Subset(void)
{
   p_node_list.clear();
   delete_matrix<dist_t>(p_node_dist, p_node_dist_size);
   delete_matrix<unsigned int>(p_node_edgeDist, p_node_dist_size);

   p_neighbor_list.clear();
}

bool Subset::isContain(node_t a_node)
{
   for(unsigned int pos = 0; pos < p_node_list.size(); ++pos)
      if(p_node_list[pos] == a_node)
         return true;

   return false;
}

bool Subset::isNeighbor(node_t a_node)
{
   for(unsigned int pos = 0; pos < p_neighbor_list.size(); ++pos)
        if(p_neighbor_list[pos] == a_node)
           return true;

   return false;
}

bool Subset::isConnected(void)
{
   for(unsigned int i = 0; i < p_node_dist_size; ++i)
      for(unsigned int j = 0; j < p_node_dist_size; ++j)
         if(p_node_edgeDist[i][j] >= p_input_data->bigM())
            return false;
   return true;
}

bool Subset::isConvex(void)
```

```
{
   if(p_node_dist_size == 0) return true;
   node_t nodeA, nodeB;
   for(unsigned int i = 0; i < p_node_dist_size-1; ++i)
      for(unsigned int j = 1; j < p_node_dist_size; ++j)
      {
         nodeA = p_node_list[i];
         nodeB = p_node_list[j];
         if(p_node_edgeDist[i][j] > p_input_data->edgeDistance(nodeA, nodeB))
            return false;
      }
   return true;
}

void Subset::add(node_t a_node, bool force)
{
   //If the cell is not already present in the subset
   if(!isContain(a_node))
   {
      //Include cell to the subset
      p_node_list.push_back(a_node);
      //Update area
      p_area += p_input_data->area(a_node);
      //Update crime risk
      p_crime += p_input_data->crime(a_node);
      //Update the neighbor list
      updateNeighborList(a_node, force);
      //Update distance matrix
      updateDistanceMatrix();
      //Update diameter
      updateDiameter();
      //Update center of gravity
      updateCenterMass();
   }
   else
   {
      cerr << "ERROR in Subset::add : argument already belongs to the subset." << endl;
      exit(EXIT_FAILURE);
   }
}

void Subset::remove(node_t a_node)
{
   if(isContain(a_node))
   {
      bool found = false;
      //Remove node from the subset
      for(unsigned int pos = 0; !found && pos < p_node_list.size(); ++pos)
            if(p_node_list[pos] == a_node)
            {
               found = true;
```

```
            p_node_list.erase(p_node_list.begin()+pos);
          }

      //Update area
      p_area -= p_input_data->area(a_node);
      //Update crime
      p_crime -= p_input_data->crime(a_node);
      //Update the neighbor list
      computeNeighborList();
      //Update distance matrix
      updateDistanceMatrix();
      //update Diameter
      updateDiameter();
      //update Center of Gravity
      updateCenterMass();
   }
   else
   {
      cerr << "ERROR in Subset::remove : argument doesn't belong to the subset." << endl;
      exit(EXIT_FAILURE);
   }

}

void Subset::updateCenterMass(void)
{
   dist_t best_maxDist = numeric_limits<dist_t>::max();
   dist_t best_sumDist = numeric_limits<dist_t>::max();

   dist_t tmp_maxDist, tmp_sumDist;
   for(unsigned int sel = 0; sel < p_node_dist_size; ++sel)
   {
      tmp_sumDist = tmp_maxDist = 0;
      for(unsigned int pos = 0; pos < p_node_dist_size; ++pos)
      {
         tmp_sumDist += p_node_dist[sel][pos] * p_input_data->crime(p_node_list[pos]);
         if(p_node_dist[sel][pos] > tmp_maxDist)
            tmp_maxDist = p_node_dist[sel][pos] * p_input_data->crime(p_node_list[pos]);
      }

      if(tmp_maxDist < best_maxDist ||
         (tmp_maxDist == best_maxDist && tmp_sumDist < best_sumDist)){
         best_maxDist = tmp_maxDist;
         best_sumDist = tmp_sumDist;
         p_center_mass = p_node_list[sel];
      }
   }
}

void Subset::computeNeighborList(void)
{
```

```
   node_t node, tmp_node;
   bool found;

   //Empty the list
   p_neighbor_list.clear();

   //Loop on all the elements in the subset
   for(unsigned int sel = 0; sel < p_node_list.size(); ++sel)
   {
      node = p_node_list[sel];

      //Including all the nodes neighboring the current one.
      for(unsigned int pos = 0; pos < p_input_data->numEdges(node); pos++)
      {
         tmp_node = p_input_data->edgeByPos(node, pos);

         //If the node is included and doesn't already belong to the subset
         if(!isContain(tmp_node))
         {
            //Check if the node has already been included in the neighbor
            found = false;
            for(unsigned int tmp = 0; !found && tmp < p_neighbor_list.size(); ++tmp)
               found = (p_neighbor_list[tmp] == tmp_node);

            //If the node has never been included in the neighbor
            if(!found)
               //Add the node to the neighbor
               p_neighbor_list.push_back(tmp_node);
         }
      }
   }
}

void Subset::updateDiameter(void)
{
   p_diameter = 0;

   for(unsigned int i = 0; i < p_node_dist_size-1; ++i)
      for(unsigned int j = i; j < p_node_dist_size; ++j)
         if(p_node_dist[i][j] > p_diameter)
            p_diameter = p_node_dist[i][j];
}

void Subset::updateDistanceMatrix(void)
{
   //Erase the previous distance matrices
   delete_matrix<dist_t>(p_node_dist, p_node_dist_size);
   p_node_dist = NULL;
   delete_matrix<unsigned int>(p_node_edgeDist, p_node_dist_size);
   p_node_edgeDist = NULL;
```

```
   //Populate the matrix
   p_node_dist_size = p_node_list.size();
   p_node_dist = new_matrix<dist_t>(p_node_dist_size, p_node_dist_size);
   p_node_edgeDist = new_matrix<unsigned int>(p_node_dist_size, p_node_dist_size);
   for(unsigned int i = 0; i < p_node_dist_size; ++i)
   {
      for(unsigned int j = 0; j < p_node_dist_size; ++j)
         if(i == j){
            p_node_dist[i][j] = 0;
            p_node_edgeDist[i][j] = 0;
         } else if(p_input_data->isEdge(p_node_list[i], p_node_list[j])){
            p_node_dist[i][j] = p_input_data->distance(p_node_list[i], p_node_list[j]);
            p_node_edgeDist[i][j] = 1;
         } else{
            p_node_dist[i][j] = p_input_data->bigM();
            p_node_edgeDist[i][j] =(unsigned int) p_input_data->bigM();
         }
   }

   //All-pairs shortest path matrix
   FloydWarshall(p_node_dist, p_node_dist_size);
   FloydWarshall(p_node_edgeDist, p_node_dist_size);
}

node_t Subset::get(unsigned int a_pos)
{
   if(a_pos < p_node_list.size()) return p_node_list[a_pos];
   else
   {
      cerr << "ERROR in Subset::get : argument out of range." << endl;
      exit(EXIT_FAILURE);
   }
}

unsigned int Subset::numElements(void)
{
   return p_node_list.size();
}

void Subset::updateNeighborList(node_t a_node, bool force)
{
   node_t tmp_node;
   bool found;

   //Eliminating the new node from the neighbor list.
   found = force;
   for(unsigned int pos = 0; !found && pos < p_neighbor_list.size(); ++pos)
      if(p_neighbor_list[pos] == a_node)
      {
         found = true;
         p_neighbor_list.erase(p_neighbor_list.begin()+pos);
```

```
      }
   if(!found && p_neighbor_list.size() > 0)
   {
      cerr << "ERROR in Subset::updateNeighborList : argument node does not belong to the neighbor
           list." << endl;
      exit(EXIT_FAILURE);
   }

   //Including all the nodes neighboring the current one.
   for(unsigned int pos = 0; pos < p_input_data->numEdges(a_node); pos++)
   {
      tmp_node = p_input_data->edgeByPos(a_node, pos);

      //If the node is included and doesn't already belong to the subset
      if(!isContain(tmp_node))
      {
         //Check if the node has already been included in the neighbor
         found = false;
         for(unsigned int tmp = 0; !found && tmp < p_neighbor_list.size(); ++tmp)
            found = (p_neighbor_list[tmp] == tmp_node);

         //If the node has never been included in the neighbor
         if(!found)
            //Add the node to the neighbor
            p_neighbor_list.push_back(tmp_node);
      }
   }
}

//Returns the number of nodes in the neighbor list.
unsigned int Subset::numNeighElements(void)
{
   return p_neighbor_list.size();
}

//Returns a node of the neighbor list at a specific position.
node_t Subset::getNeighbor(unsigned int a_pos)
{
   if(a_pos < p_neighbor_list.size()) return p_neighbor_list[a_pos];
   else
   {
      cerr << "ERROR in Subset::getNeighbor : argument out of range." << endl;
      exit(EXIT_FAILURE);
   }
}

crime_t Subset::crime(void)
{
   return p_crime;
}
```

```
area_t Subset::area(void)
{
   return p_area;
}

dist_t Subset::diameter(void)
{
   return p_diameter;
}

unsigned int Subset::support(void)
{
   return p_support;
}
void Subset::setSupport(unsigned int a_support)
{
   p_support = a_support;
}

node_t Subset::centerMass(void)
{
   return p_center_mass;
}

double Subset::calculateWLoad(void)
{
   wload_t tmp_load = 0;

   //First attribute: Demand ratio
   tmp_load += Parameters::WDEMAND * p_crime / p_input_data->totalCrime();

   //Second attribute: Area ratio
   tmp_load += Parameters::WAREA * p_area / p_input_data->totalArea();

   //Third attribute: Diameter ratio
   tmp_load += Parameters::WDIAMETER * p_diameter / p_input_data->diameter();

   //Fourth attribute: Support ratio
   tmp_load += Parameters::WSUPPORT * (p_input_data->numSubsets() - 1 - p_support) /
        (p_input_data->numSubsets() - 1);

   return tmp_load;
}
```

A.8. File: myalgorithm.hpp

This file contains the definition of the Floyd-Warshal algorithm for the computation of the shortest-path matrix.

myalgorithm.hpp

```
#if !defined MYALGORITHM_HPP
#define MYALGORITHM_HPP

#include "mymemory.hpp"

using namespace std;

template <typename T>
void FloydWarshall(T **a_matrix, unsigned int a_nodes)
{
   //T **matrix = copy_matrix(a_matrix, a_nodes, a_nodes);
   for(unsigned int k = 0; k < a_nodes; ++k)
      for(unsigned int i = 0; i < a_nodes - 1; ++i)
         for(unsigned int j = i + 1; j < a_nodes; ++j)
            if(a_matrix[i][k] + a_matrix[k][j] < a_matrix[i][j])
             a_matrix[i][j] = a_matrix[j][i] = a_matrix[i][k] + a_matrix[k][j];
}

#endif
```

A.9. File: mymemory.hpp

mymemory.hpp provides basic procedures for the management of matrices data structures.

mymemory.hpp

```
#if !defined MYMEMORY_HPP
#define MYMEMORY_HPP
#include <cstring>


template <typename T>
T** new_matrix(unsigned int a_row, unsigned int a_col)
{
   if(a_row <= 0 || a_col <= 0)
      return NULL;

   T** matrix = new T*[a_row];
```

```
   for(unsigned int i = 0; i < a_row; ++i){
      matrix[i] = new T[a_col];
      for(unsigned int j = 0; j < a_col; ++j){
         matrix[i][j] =(T) 0;
      }
   }

   return matrix;
}

template <typename T>
void delete_matrix(T** a_matrix, unsigned int a_row)
{
   if(a_matrix != NULL){
      for(unsigned int i = 0; i < a_row; ++i){
         if(a_matrix[i] != NULL) delete[] a_matrix[i];
      }
      delete[] a_matrix;
   }
}

template <typename T>
T** copy_matrix(T** a_matrix, unsigned int a_row, unsigned int a_col)
{
   if(a_matrix == NULL || a_row <= 0 || a_col <= 0)
      return NULL;

   T** matrix = new_matrix<T>(a_row, a_col);
   for(unsigned int i = 0; i < a_row; ++i){
      memcpy(matrix[i], a_matrix[i], a_col*sizeof(T));
   }

   return matrix;
}

#endif
```

APPENDIX B

SAMPLE PROBLEM INSTANCE

We now present a sample problem instance for the MC-PDP. Specifically, this example refers to the night shift of Saturday 10/15/2011, from 10 PM to 8 AM. An instance is comprised of the following parts:

- Number of nodes.
- Risk associated with each node. In the sample instance, it represents the number of expected thefts that will occur in the corresponding area during the considered shift.
- Area associated with each node. In the sample instance, it is calculated as the sum of the lengths (in meters) of the streets that belong to the considered area.
- Number of edges.
- List of edges and associated distance as a triplet {source, destination, distance}. In the instance the distances are expressed in meters.

```
111

1.847435032 0.1556274449 0.622201076 1 0.5210441332 0.681074225 1.4952551484 0.9773838127 0
    0.5903392415 0.5628374046 0 0 0 0 0 0 0.5211234734 0.5834516704 0.4361932303 0.2348564477
    1.3596163852 2.7351950533 0.6646511965 1.7169609736 0 0 0 0 0 0 0 0 0 0 0.0013616558 0 0
```

```
    0.1418114507 0.2599278718 0.2908085259 0 0.325805989 0.4017393225 0 1.2495583367 1.247059666
    0 0 0 0 0 0.5183809999 0.5462574949 0.420142739 0.2719482174 0 0 0 0.2954159984 2.2754077918
    1.8064755012 3.6682885544 0 2.610233578 3 1.4335569947 1.0294728075 0.5448328366
    0.9278174276 0.7160613989 1.7108410694 1.4897414961 1.1752472477 2.1759956517 2.0027277849
    0.0026898746 0.0026898746 0 0 0.0016987842 0.115539476 0.0035917092 0 0 0 0 0.3760499461
    0.2854458854 0.1786589192 0 0.2216245926 0 0 0 0 0.5116581304 1.1606141128 0.357061161
    0.2761369351 0 0 0.2667712853 0.357676439 3.8648130035 0.1427225802 1.4823337139
    2.6810386478 4.5424046962 1.3736329756 2.2426371065

10637.8872114451 1173.2769893873 2758.8700868319 1787.9890877009 2461.1753810222 1090.853926134
    1483.5956187032 3456.290740624 2100.6547195789 996.5333417578 1304.411931992 1156.8964938761
    1946.7897353357 1208.0347571006 2444.2107729082 623.9632981413 1337.1321940978
    1428.2902193829 1170.1374591127 1106.4392538321 836.4159434318 1411.8715397772
    1583.9677789648 1185.2191479776 1647.0286264933 945.8844274653 793.9633631986
    1176.2236941508 1596.4448672656 848.6033507724 1021.144546867 1125.4345941882 894.1678747523
    721.270369998 607.4444014646 1638.7233120348 444.5422139508 828.9932405194 756.7402334221
    567.5827150186 1220.3438557124 1080.7024206932 1331.4297306652 668.9632511685 712.4229946604
    1049.2687601145 870.444985404 1101.3181423807 713.9108625644 688.4900578513 478.9529935614
    1126.6314459422 1415.5398425038 1339.705903453 837.0835942753 1813.4933502728 985.0861576098
    1604.0825864119 953.5708003785 1066.573921927 1474.7604162864 1463.2195243717
    1502.0734020987 1288.282106473 4560.8872801325 2611.4997715881 2639.6419417113
    1282.271141329 1298.5228441698 1346.6817891962 1570.0774756507 1455.084327801
    1782.0840480148 2825.7590949842 2462.2665830782 1876.0283946461 858.4741616318
    885.8925321892 788.8190543854 504.8093999718 993.1836208315 1527.1380441207 998.2037957052
    889.421191316 1155.2865292616 911.5433132603 1045.8751854988 1192.2908098112 867.0416009103
    1555.4527254373 1207.9802554586 1076.1166227739 1647.6272092661 643.0536522822
    1316.7260209498 1905.1267125181 821.4243363018 1470.0304931748 1140.4426361983
    2058.1429140528 924.5701697846 1327.026045963 2037.6467818207 1961.8213548161
    3589.6338241939 1175.6238536709 2802.1082634262 2047.4937094759 2979.9093937573
    1400.3307655069 2992.9475203784

276

1 2 646.4820551767
1 4 481.3878312856
1 5 461.1720019008
1 6 439.0575877958
1 8 519.324813152
1 15 586.2965591151
2 3 205.8168679553
2 4 290.3870591101
2 100 228.6375224773
3 100 283.6679667638
3 103 226.4965892555
3 105 333.0346968893
3 5 391.9884993202
3 4 214.874188596
4 5 214.3471955242
5 105 455.5710778691
5 111 366.6435770383
5 110 265.7709295427
```

```
5 6 140.2422051988
6 110 304.9873120699
6 7 189.9294667145
6 8 209.6537023257
7 109 200.8761716717
7 8 162.7903871665
8 109 358.8668602735
8 10 289.2399471467
8 9 188.0350047585
8 14 180.6860262938
9 10 161.5085529271
9 12 233.1856574503
9 13 252.7902638656
9 14 196.3174790692
10 109 333.2787261526
10 23 330.8116984716
10 22 162.3059284584
10 11 267.7845890205
11 21 165.5821135919
11 20 153.2746266649
11 18 264.8507001939
11 17 285.3897951831
11 12 126.7603011744
12 17 212.1709992221
12 13 163.5441113246
13 14 197.2520751395
13 17 250.8356315793
13 15 152.078845272
14 15 231.1232154167
15 17 329.894794384
15 16 356.4069014923
16 17 171.0589935458
16 18 278.5029316041
17 18 153.0277791679
18 19 194.2154483222
19 20 168.455120456
19 29 203.0157536088
20 21 170.4959926642
20 28 149.8048391736
20 29 245.7899128206
21 22 235.8352206241
21 28 107.5096938582
22 23 206.8585528052
22 25 168.1013193459
22 28 304.9013522838
23 24 278.729386583
23 25 190.8879140251
23 109 241.078705742
24 61 122.1318308699
24 54 388.4500964391
24 41 102.4352078293
```

```
24 25 301.480044472
25 41 285.5694532194
25 43 218.9089081755
25 31 244.593293541
25 26 146.9885807274
26 31 223.1073169655
26 27 129.0737733357
26 28 198.0390239936
27 31 199.4564428057
27 30 58.3159331727
27 28 148.3468804664
28 29 240.9805906922
29 30 243.5633398556
29 33 215.3367479173
29 34 210.3001858967
29 36 387.8316420411
30 31 198.6449699537
30 32 200.8803023129
30 33 93.262056887
31 43 182.8908297802
31 40 67.6734301078
31 39 222.4288454119
31 32 146.5627636574
32 39 167.081826059
32 35 121.3908191215
32 34 185.0170850582
32 33 157.2546717475
33 34 105.3690782354
34 36 187.3534446983
35 39 141.4402819544
35 36 119.9775343063
36 39 257.1949481435
36 37 186.4828965357
37 38 116.9556824954
38 39 164.1127644613
38 46 209.0343819419
38 49 145.163711969
38 50 147.2530570907
38 51 127.1045401344
39 40 222.747987693
39 44 157.2112433606
39 46 135.8612044331
40 43 117.8698217849
40 44 120.5881932569
41 54 331.9063610081
41 42 194.8898922863
41 43 134.3946237642
42 54 150.3277197067
42 47 155.5828865089
42 46 227.0824056997
42 45 132.1979614905
```

```
42 43 242.6258649459
43 44 147.4952784438
43 45 192.5829441477
44 45 108.6424325347
44 46 135.6730320697
45 46 110.1471905817
46 47 134.9909545372
46 49 152.6081391632
47 54 180.5172144654
47 48 145.3036398797
48 54 229.1635113835
48 53 260.8178200641
48 52 188.219827619
48 50 196.7457648927
48 49 168.4288100713
49 50 78.1074161957
50 52 237.9839594228
50 51 91.200120063
51 52 294.5053478463
52 53 207.9871317684
53 54 335.7149342657
53 55 178.9114886595
53 56 160.6491303737
54 60 139.5738097133
54 55 198.4176527865
55 56 128.2431531977
55 60 212.1799533904
56 57 145.1286020682
56 60 315.5163697246
57 58 137.9204149293
57 60 249.7348034094
58 64 174.021048116
58 59 241.819501206
58 60 195.3506265738
59 60 172.390571984
59 61 221.399366416
59 62 152.4996215078
59 64 209.337528397
60 61 312.852004482
61 63 208.6162975407
61 62 137.748627377
62 63 175.5818783157
62 65 305.8674620464
62 64 314.3664518636
63 107 182.5247715
63 65 344.7811009549
64 65 290.3952049198
65 107 372.1609255761
65 106 381.8907226312
65 67 370.4560298529
65 66 688.1630934292
```

```
66 74 280.515078596
66 71 253.3645009578
66 70 251.8980062807
66 67 410.1075241233
67 70 214.8853988638
67 69 182.2829137927
67 68 298.0543451015
67 106 282.0015932388
68 104 251.2239088984
68 83 123.546290144
68 72 230.8886281308
68 69 122.7700055184
68 106 328.4287270392
69 72 229.5809601962
69 70 189.1951953478
70 72 210.1222043652
70 71 201.1656723018
71 72 164.8581965998
71 73 224.2084603822
71 74 251.0362217065
72 73 165.9689155772
72 82 305.9463344119
72 83 271.6984693737
73 74 388.3438210282
73 75 215.7885298985
73 82 262.0238621982
74 75 283.3769922776
75 76 227.2703643265
75 82 451.1434952119
76 77 147.7967042809
76 81 265.6274385267
76 82 324.0238911463
77 78 186.4155841634
78 79 194.9596992752
78 80 121.0160389758
78 81 153.8737989471
79 80 84.8550509526
79 90 204.6899001232
79 91 162.5830055735
80 90 157.7453893728
81 82 245.9260714456
81 89 121.0402924611
81 90 205.278016978
82 83 217.1560983402
82 84 218.9108702909
82 85 186.9132164052
82 89 159.0263503603
83 84 131.5831795589
83 104 230.4840748615
84 85 134.3865505995
84 104 217.8951910379
```

```
84 102 153.4854157177
85 89 201.4147019479
85 86 100.9254915409
85 102 197.6181934849
86 87 157.7254105234
86 88 135.6819313517
86 89 173.9351407835
86 101 221.1227006513
87 88 125.8362835552
87 101 143.6786647293
87 98 179.9624988143
88 90 134.7183888318
88 89 213.2137653261
88 98 247.7877178949
89 90 211.8456077149
90 91 243.7738806913
90 92 233.0897550153
91 94 148.2911850778
91 93 202.5617193749
91 92 193.1359274635
92 93 141.4274377387
92 97 159.1439845842
92 98 164.308852143
93 94 146.073367511
93 96 200.7407348029
93 97 142.5780855721
94 95 307.8283580361
95 96 252.947943015
96 97 173.1046541144
96 98 306.5311606915
96 99 262.419347052
97 98 138.3441260338
98 99 216.0112691581
98 100 238.6189559171
99 100 231.3808633981
100 101 180.4033046033
100 103 282.9772753442
101 102 208.0969760746
101 103 223.5274826566
102 103 163.9039458959
102 104 207.9353792749
103 104 279.4762736117
103 105 264.1958990449
104 105 265.1177438164
105 106 371.3249373993
105 107 400.9021230293
105 111 166.8739686659
106 107 267.3883538215
107 108 245.3931701596
107 111 344.3389045471
108 109 206.5778347614
```

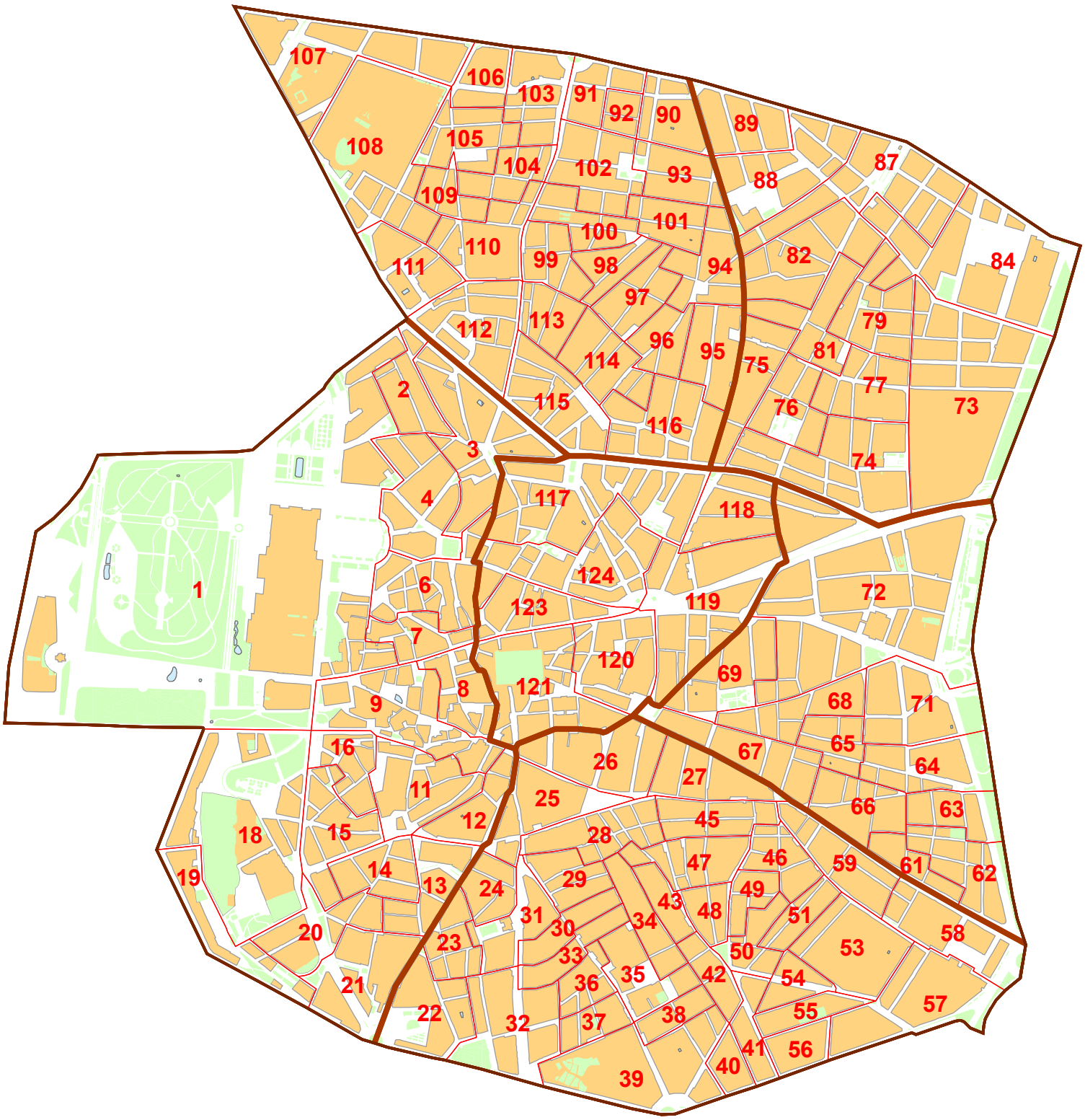
```
108 110 250.7839254911
109 110 193.3128827599
110 111 164.6725934034
```

APPENDIX C

MAP OF THE CENTRAL DISTRICT OF MADRID

We now provide a map of the Central District of Madrid (in Spanish) illustrating the partition in census districts. Thick lines delimit the six neighbors comprising the Central District, while the thin lines demarcate the census districts. Each census district is identified by a number. It is important to notice that some of the numbers are missing (i.e., 5, 10, 17, 44, 52, 60, 70, 78, 80, 83, 85, 86, 122). Thus, the total number of census districts is 111, although the highest numbered district is the 124th.

Distrito 01 - Centro



Barrios

11 - Palacio

12 - Embajadores

13 - Cortes

14 - Justicia

15 - Universidad

16 - Sol