Problema de la mochila con capacidad variable

Universidad Miguel Hernández

Facultad de Ciencias Sociales y Jurídicas

Grado en Estadística Empresarial

Trabajo de Fin de Grado

Autor: Alejandro Moya Martínez

Tutor: Juan Fancisco Monge Ivars

Convocatoria: Febrero 2017

Índice general

Capítulo 1

Introducción

La investigación de operaciones (IO) se inició en Inglaterra durante la Segunda Guerra Mundial, cuando un equipo de científicos empezó a tomar decisiones respecto a la mejor utilización del material militar. Todas estas ideas se adaptaron para mejorar la eficiencia y productividad en la vida cotidiana [1].

La IO busca cuantificar un aspecto mejorable, donde la mejora va estar condicionada por la toma de decisiones. Se pretende crear un modelo matemático que represente la realidad del problema y así poder obtener una solución factible y real para dicho problema.

En este proyecto abarcaremos un modelo de optimización dinámica y combinatoria llamado Problema de la Mochila, "Knapsack Problem"(KP). Es uno de los 21 problemas NP-completos de Richard Karp, lo que quiere decir, de los 21 problemas no deterministas con dificultad para resolver. La formulación del problema es sencilla, pero la dificultad llega al intentar resolverlo computacionalmente.

Este tipo de problema lo clasifican como problema de programación lineal entera-mixta, aunque nuestro trabajo abarcará un modelo de programación cuadrática y exponencial con variables binarias.

Un ejemplo de una situación económica real para el problema de la mochila, el de cargamento de paquetes en una línea aérea. El operario obtiene una lista de paquetes de los clientes, él debe de decidir que paquetes debe cargar en el avión. Esta lista viene dada por unos pesos y una tasa de pago por transporte del paquete, es decir, nos proporciona un beneficio y nos repercute un coste.

La primera condición que impondremos, es que beneficios y pesos deben ser positivos, y estos se denotarán por dos vectores continuos, (Vector: b) y (Vector: p).

A continuación se presentará el problema de la mochila clásico, es el modelo del que se partirá para formular nuestro proyecto , en el se asumirá que los beneficios y los pesos son no negativos como hemos puntualizado anteriormente. Además tendremos un parámetro de capacidad c, que vendrá dado, para no superarlo [1].

Los parámetros y conjuntos en nuestro modelo son los siguientes.

Conjuntos:

$I = \{1, ..., n\}$ conjunto de productos o paquetes a seleccionar.

Parámetros:

b_i= Beneficio que repercute al incorporar el producto i en la solución, $\forall i \in I$.

p_i= Peso que repercute al incorporar el producto i en la solución, $\forall i \in I$.

Variables: x_i= Variable binaria que toma valor 1 si seleccionamos el producto i y toma valor 0 en caso contrario $\forall i \in I$.

La formulación del problema de la mochila quedaría de la siguiente forma:

$$(KP) \equiv \quad Max \textstyle\sum_{i=1} b_i x_i \tag{1.1}$$

$$s.a \textstyle\sum_{i=1} p_i x_i \leq c \tag{1.2}$$

$$x_i \in \{0, 1\} \qquad \forall i = 1, ..., I. \tag{1.3}$$

El problema (KP) es el modelo de programación entera no trivial más simple con variables binarias, una sola restricción y los coeficientes positivos. Sin embargo, al incorporar la condición de integridad (1.3) al programa lineal simple (1.1) - (1.2) ya coloca (KP) en la clase de problemas "difíciles"[1].

Nuestro proyecto estará basado en el estudio del problema, dependiendo de la función de pesos que queramos imponer. Lo que conseguimos en este tipo de problemas es minimizar una función de pesos, al mismo tiempo que maximizamos el beneficio. Además, impondremos restricciones como puede ser la selección del números de paquetes o no superar un coste determinado.

Capítulo 2

Problema de la mochila con capacidad variable

En primer lugar mostraremos la forma que tiene el problema de la mochila con capacidad variable para cualquier función de pesos, $f(P)$.

$$Max \quad \sum_{i=1} b_i x_i - \lambda f(P) \tag{2.1}$$

$$s.a. \qquad P = \sum_{i=1} p_i x_i \tag{2.2}$$

$$x_i \in \{0,1\} \qquad \forall i = 1, ..., I. \tag{2.3}$$

donde P es el peso de la mochila en la solución del problema.

Si consideramos que $f(P)$ es una función convexa del peso de la mochila, el problema (2.1)-(2.3) se corresponde a un problema de optimización convexa con variables 0-1. También tendremos un parámetro (en la función objetivo), λ que es el coeficiente de adversión al peso, es decir, la penalización por incluir una unidad adicional del peso en la mochila.

En el presente trabajo analizaremos 3 casos diferentes en los que se consideran 3 funciones convexas distintas para $f(P)$:

1. Función lineal, $f(P) = \sum_{i=1} p_i x_i$.

2. Función cuadrática, $f(P) = (\sum_{i=1} p_i x_i)^2$.

3. Función exponencial, $f(P) = e^{(\sum_{i=1} p_i x_i)}$.

2.1. Función de pesos lineal

Si la función del peso de la mochila, $f(P)$ es lineal, el problema de la mochila con capacidad variable vendrá dado por el siguiente modelo de programación lineal con variables {0,1}:

$$(KPVC - l) \equiv \quad Max \sum_{i=1} b_i x_i - \lambda \sum_{i=1} p_i x_i \tag{2.4}$$

$$s.a. \; x_i \in \{0,1\} \qquad \forall i \in I \tag{2.5}$$

Veamos que la solución del problema de la mochila con capacidad variable para el caso de que la función f(P) es lineal tiene solución trivial. Podemos simplificar nuestro modelo sacando factor común la variable x_i:

$$(KPVC - l) \equiv \quad Max \sum_{i=1} (b_i - \lambda p_i) x_i \tag{2.6}$$

$$s.a. \; x_i \in \{0,1\} \qquad \forall i \in I \tag{2.7}$$

Con esta expresión se puede comprobar que las soluciones optimas es incorporar a la mochila aquellos productos, donde $b_i - \lambda p_i \geq 0$. Evidentemente la solución del problema depende del parámetro λ. Veamos mediante un ejemplo como interviene λ en la solución. Suponiendo los datos de b={1,2,4} y p= {2,3,5}, la función a maximizar será la siguiente:

$$z = max\ x_1 + 2x_2 + 4x_3 - \lambda(2x_1 + 3x_2 + 5x_3) = \quad (2.8)$$

$$= (1 - 2\lambda)x_1 + (2 - 3\lambda)x_2 + (4 - 5\lambda)x_3. \quad (2.9)$$

Comenzaremos comprobando los casos extremos, es decir, el caso de no seleccionar ningún paquete, y el caso de seleccionar todos ellos, estos casos se corresponde a los valores de $\lambda = +\infty$ y $\lambda = 0$ respectivamente:

1. Si $\lambda = +\infty \longmapsto x = (0, 0, 0) \longmapsto z^* = 0$

2. Si $\lambda = 0 \longmapsto x = (1, 1, 1) \longmapsto z^* = 7$

3. Si $\lambda \in (0, 1/2) \longmapsto z* = (1 - 2\lambda) + (2 - 3\lambda) + (4 - 5\lambda) = 7 - 10\lambda$

   Podemos decir que la solución será de x=(1,1,1).

4. Si $\lambda \in (1/2, 2/3) \longmapsto z* = (2 - 3\lambda) + (4 - 5\lambda) = 6 - 8\lambda$

   La solución óptima x=(0,1,1).

5. Si $\lambda \in (2/3, 4/5) \longmapsto z* = 4 - 5\lambda$

   La solución óptima x=(0,0,1).

6. Si $\lambda \geq 4/5 \longmapsto z* = 0$

   La solución óptima x=(0,0,0).

Como podemos apreciar, en la tabla 2.1, muestra las soluciones del problema para diferentes valores de λ. Existen combinaciones posibles que no serán soluciones del problema. Por ejemplo, (1,0,0) no es solución del problema, se puede decir, que si es ventajoso poner el primer producto de la mochila, también lo es poner el resto de artículos.

Tabla 2.1: Solución f(P)lineal

Solución óptima	λ
(0,0,0)	$(+\infty$, 4/5)
(0,0,1)	[4/5,2/3)
(0,1,1)	[2/3,1/2)
(1,1,1)	[1/2,0]

2.2. Caso con la función de pesos cuadrática

Si la función del peso de la mochila, $f(P)$ es cuadrático, el problema de la mochila con capacidad variable tendrá la siguiente forma :

$$(KPVC-c) \equiv \quad Max \sum_{i=1} b_i x_i - \lambda(\sum_{i=1} p_i x_i)^2 \tag{2.10}$$

$$x_i \in \{0,1\} \qquad \forall i \in I \tag{2.11}$$

Suponemos los mismos datos que en el caso anterior.

$$z = max\ x_1 + 2x_2 + 4x_3 - \lambda(2x_1 + 3x_2 + 5x_3)^2 \tag{2.12}$$

En la tabla 2.2, se puede ver para los diferentes λ que solución es la idónea para el problema planteado anteriormente.

Como se puede apreciar, en este caso se obtiene dos nuevas soluciones del problema (2.10)-(2.11). Actualmente, la mayoría de optimizadores del estado del arte resuelven problemas de optimización cuadrática. En este trabajo, hemos resuelto este tipo de problemas utilizando el optimizador de Cplex, integrado en las librerias en C++.

Tabla 2.2: Solución f(P) cuadrático

Solución óptima	λ
(0,0,0)	$(+\infty$, 0.25)
(1,0,0)	[0.25,0.222)
(0,1,0)	[0.222,0.16)
(0,0,1)	[0.16,0.06)
(0,1,1)	[0.06,0.027)
(1,1,1)	[0.027,0]

2.3. Caso con la función de pesos exponencial

Suponiendo que la función de pesos, f(P), es una función exponencial, el cual tendrá la siguiente forma:

$$(KPVC-e) \equiv \quad Max \sum_{i=1} b_i x_i - \lambda e^{(\sum_{i=1} p_i x_i)} \tag{2.13}$$

$$x_i \in \{0,1\} \qquad \forall i \in I \tag{2.14}$$

Aplicando los datos mencionados en la sección 2.1, podemos comprobar que el modelo que debemos resolver es el siguiente (2.13):

$$z = max\ x_1 + 2x_2 + 4x_3 - \lambda e^{(2x_1+3x_2+5x_3)} \tag{2.15}$$

La tabla 2.3, muestra la solución para diferentes valores de λ. Se obtiene una solución nueva (1,0,1) respecto al modelo cuadrático. Si nos fijamos la única solución no obtenida de todas las posibles es (1,1,0).

En definitiva la función del peso de la mochila nos proporcionará la selección de diferentes soluciones. Encontramos con un problema, ya que el modelo (2.13)-(2.14), no es capaz de ser resuelto por los optimizadores actuales, por lo que proponemos en este trabajo un algotimo para su resolución. Además demostraremos porque las

Tabla 2.3: Solución f(P) exponencial

Solución óptima	λ
(0,0,0)	$(+\infty,0.15)$
(1,0,0)	[0.15,0.075)
(0,1,0)	[0.075,0.015)
(0,0,1)	[0.015,0.001)
(1,0,1)	[0.001,0.0005)
(0,1,1)	[0.0005,0.00005)
(1,1,1)	[0.00005,0]

soluciones óptimas del modelo lineal y del modelo cuadrático óptimas son soluciones del modelo exponencial.

2.4. Solución dominada, en el sentido pareto

Como se sabe, la regla de Pareto (el más extendido criterio de eficiencia distributiva en Economía) postula que un estado de cosas "X" es superior respecto de otra "Z", sí y solo sí alguien prefiere "X" a "Z" y nadie prefiere "Z" a "X", de modo tal que "X" implica mayor utilidad que "Z". Sin embargo, "Z", afirman Kaplow y Shavell, que es la consecuencia de la aplicación, de poder otorgar peso moral independiente de la utilidad a los criterios de justicia, que tienen altos costos en utilidad, por lo tanto violentan la regla de Pareto: una de menor utilidad desde el punto de vista de las preferencias de los individuos. [5]

Resultados teóricos

Proposición 1 *Sea (B,P) una posible solución de la configuración de paquetes en la mochila, dominada en el sentido pareto, entonces no existe λ tal que (B,P) es solución del problema:*

Demostración 1 *Si (B,P) es solución dominada, entonces $\exists(\bar{B},\bar{P})$que domina a (B,P):*

$$\{\bar{B} > B, \bar{P} \leq P\} \text{ ó } \{\bar{B} \geq B, \bar{P} < P\} \tag{2.16}$$

-Suponemos $\exists\lambda^*$: (B,P) es el óptimo para el modelo (2.6)-(2.7):

$$Max \sum_{i=1} b_i x_i - \lambda \sum_{i=1} p_i x_i = B - \lambda P \tag{2.17}$$

$$B - \lambda P \geq \bar{B} - \lambda\bar{P} \tag{2.18}$$

$$(B - \bar{B}) - \lambda(P - \bar{P}) \geq 0 \tag{2.19}$$

Pero sabemos por (2.16) que $(B - \bar{B}) - \lambda(P - \bar{P}) < 0$. Por lo tanto (B,P) no es solución al problema (2.6)-(2.7).

Proposición 2 *Sea (B,P) una posible solución de la configuración de paquetes en la mochila, dominada en el sentido Pareto, entonces no existe un λ tal que (B,P) sea solución del problema (2.20):*

$$(KPVC - c) \equiv max \sum_{i=1} b_i x_i - \lambda(\sum_{i=1} p_i x_i)^2 \tag{2.20}$$

Demostración 2 *Si (B,P) es solución dominada, entonces $\exists(\bar{B},\bar{P})$que domina a (B,P):*

-Suponemos $\exists\lambda^*$: (B,P) es el óptimo para el problema (2.20):

$$Max \sum_{i=1} b_i x_i - \lambda(\sum_{i=1} p_i x_i)^2 = B - \lambda P^2 \tag{2.21}$$

$$B - \lambda^* P^2 \geq \bar{B} - \lambda \bar{P}^2 \tag{2.22}$$

$$(B - \bar{B}) - \lambda(P^2 - \bar{P}^2) \geq 0 \tag{2.23}$$

Pero sabemos por (2.16) que $(B - \bar{B}) - \lambda(P^2 - \bar{P}^2) < 0$. Por lo tanto (B,P) no puede ser solución al problema (2.20).

Proposición 3 *Sea (B,P) una posible solución de la configuración de paquetes en la mochila, dominada en el sentido Pareto, entonces no existe un λ tal que (B,P) sea solución del problema exponencial:*

$$(KPVC - e) max \sum_{i=1} b_i x_i - \lambda e^{(\sum_{i=1} p_i x_i)} \tag{2.24}$$

Demostración 3 *Si (B,P) es solución dominada, entonces $\exists(\bar{B}, \bar{P})$que domina a (B,P):*

-Suponemos $\exists\lambda^*$: (B,P) es el óptimo para el problema (2.24):

$$Max \sum_{i=1} b_i x_i - \lambda e^{(\sum_{i=1} p_i x_i)} = B - \lambda e^P \tag{2.25}$$

$$B - \lambda^* e^P \geq \bar{B} - \lambda e^{\bar{P}} \tag{2.26}$$

$$(B - \bar{B}) - \lambda(e^P - e^{\bar{P}}) \geq 0 \tag{2.27}$$

Pero sabemos por (2.16) que $(B-\bar{B})-\lambda(e^{P}-e^{\bar{P}})<0$. Por esta causa aseguramos que (B,P) no es solución al problema del modelo exponencial (2.24).

2.5. Soluciones heredadas dependiendo de $f(P)$

Ahora mostraremos la comprobación de los ejemplos, de porque las soluciones de los modelos mas sencillos están en los modelos mas complejos. Comenzaremos hablando de la solución obtenida en el modelo lineal $(KPVC-l)$, será solución en el modelo cuadrático $(KPVC-c)$, y continuaremos con la demostración de toda solución del modelo cuadrático $(KPVC-c)$, son soluciones del modelo exponencial $(KPVC-e)$.

Proposición 4 *Una solución del problema lineal* $(KPVC-l)$ *es solución del problema cuadrático* $(KPVC-c)$.

$$B-\lambda P \geq B^{*}-\lambda P^{*} \quad \forall (B^{*},P^{*}) \tag{2.28}$$

Demostración 4 *Suponemos que* (B,P) *es solución del problema lineal, entonces podemos decir que:*

$$B-\lambda P \geq B^{*}-\lambda P^{*} \quad \forall (B^{*},P^{*}) \tag{2.29}$$

Queremos probar, que $\exists\ \lambda'$ *donde,*$(B.P)$ *es solución del problema cuadrático* $(KPVC-c)$*, es decir,*

$$B-\lambda P^{2} \geq B^{*}-\lambda P^{*2} \quad \forall (B^{*},P^{*}) \tag{2.30}$$

Demostraremos que las soluciones del modelo lineal serán mayores o iguales a 0, partiendo de un modelo cuadrático.

$$(B - B^*) - \lambda'(P^2 - P^{*2}) = (B - B^*) - \lambda'(P - P^*)(P + P^*) = \tag{2.31}$$

$$= (B - B^*) - \lambda'(P - P^*)(P + P^*) = \tag{2.32}$$

$$= (B - B^*) - \lambda(P + P^*)(P - P^*) \geq 0 \tag{2.33}$$

Sabiendo que $\lambda = \lambda'(P + P^*)$, *obtenemos que son positivas las soluciones, es decir, que las soluciones del modelo (KPVC-l) estarán integradas en las soluciones del modelo (KPVC-e).*

A continuación veremos el segundo caso, que las soluciones del modelo cuadrático$(KPVC-c)$, son soluciones del modelo exponencial $(KPVC - e)$.

Proposición 5 *Una solución del problema cuadrático es solución del problema exponencial.*

Demostración 5 *Si (B,P) es solución del problema cuadrático* $(KPVC - c)$ *es solución del problema exponencial* $(KPVC - e)$, *es decir:*

- Suponiendo que (B, P) *es solución del problema cuadrático* $(KPVC - c)$, *entonces :*

$$B - \lambda P^2 \geq B^* - \lambda P^2 \ \forall (B^*, P^*) \tag{2.34}$$

Queremos probar, que $\exists \ \lambda'$ *donde,(B.P) es solución del problema exponencial* $(KPVC - e)$, *es decir:*

$$(B - B^*) - \lambda'(e^P - e^{P^*}) = (B - B^*) - \lambda'(e^P - e^{P^*}) = \tag{2.35}$$

$$= (B - B^*) - \lambda' \frac{e^P - e^{P^*}}{P^2 - P^{*2}}(P^2 - P^{*2}) = \tag{2.36}$$

$$= (B - B^*) - \lambda(P^2 - P^{*2}) \geq 0 \tag{2.37}$$

Sabiendo que $\lambda = \lambda'(e^P - e^{P^*})/(P^2 - P^{*2})$*, podemos decir que es positivo ya que la función* e^P *es monótona creciente. Con ello demostramos que las soluciones del modelo cuadrático* $(KPVC - c)$*, pertenecerán al modelo exponencial* $(KPVC - e)$*.*

2.6. Método de aproximación al problema de la mochila con función de pesos exponencial

Como se ha comentado anteriormente, nuestro objetivo es maximizar una función exponencial, al obtener problemas para la resolución, se calculará una aproximación al valor exacto de la función $(f(x_1, x_2, ..., x_n))$, que tendrá forma cuadrática. Como paquetes independientes se derivada la función para obtener el resultado óptimo:

Partiendo de una función de coste con forma exponencial:

$$(KPVC - e) \equiv Max \sum_{i=1} a_i x_i - \lambda e^{\sum_{i=1} b_i x_i} \tag{2.38}$$

Suponemos un solución inicial al problema que tendrá la siguiente forma:

$$f(x^o) = \sum_{i=1} a_i {x^o}_i - \lambda e^{\sum_{i=1} b_i {x^o}_i} \tag{2.39}$$

-Podemos aproximar la función $(KPVC - e)$ mediante y su aproximación cuadrática:

$$f(x_1, x_2, ..x_n) \simeq f(x^o) + \nabla f(x^o)(x - x^o) + \frac{1}{2}(x - x^o)^t H f(x^o)(x - x^o) \tag{2.40}$$

siendo H la matriz Hessiana de f(x).

Al ordenar los términos de la expresión (una constante + el término lineal + el término cuadrático), la expresión tendrá una forma para la aproximación de $f(x)$ en el punto x^o.

$$\begin{aligned} f(x) \simeq f(x^{o}) - \nabla f(x^{o})x + \frac{1}{2}xHf(x^{o})x + \\ +(\nabla f(x^{o}) - x^{ot}Hf(x^{o}))x + \\ +\frac{1}{2}x^{t}Hf(x^{o})x^{t} \end{aligned} \tag{2.41}$$

Para simplificación la anotación pasaremos a llamar $P^{o} = \sum_{i=1} b_i {x^{o}}_i$,es decir, el peso del a mochila para la solución x^{o}.

El problema (2.38) puede aproximarse por el siguiente problema:

$$(KPVC-h) \equiv Max \sum_{i=1} a_i {x^{o}}_i - \lambda e^{P^{o}} \tag{2.42}$$

$$+\sum_{i=1}(a_i - \lambda e^{P^{o}} b_i)(-{x^{o}}_i) + \sum_{i=1}(a_i - \lambda e^{P^{o}} b_i)x_i \tag{2.43}$$

$$-\frac{\lambda e^{P}}{2}(\sum_{i=1}(b_i x_i)^2 + P^{o} + 2P^{o}x_i) \tag{2.44}$$

Si ordenamos la parte constante, lineal y cuadrática formaríamos la siguiente expresión:

$$\begin{aligned} (KPVC-h) \equiv Max - \lambda e^{P} + \lambda e^{P} P - \frac{\lambda e^{P}}{2} P^2 \\ +\sum_{i=1}(a_i - \lambda e^{P} b_i + \lambda e^{P} P b_i)x_i \\ -\frac{\lambda e^{P}}{2}\sum_{i=1}(b_i x_i)^2 \end{aligned} \tag{2.45}$$

Podemos ver como la expresión (2.45)tiene una forma cuadrática, la cual, puede resolverse mediante un optimizador, que en nuestro caso será Cplex. También nos permitirá definir un algoritmo heurístico para la resolución del problema de la mochila con capacidad variables y función del peso exponencial.

2.6.1. Algoritmo heurístico por aproximación a la función exponencial

Este algoritmo esta programado en C++, y gracias a la librerías contenidas en el sofware de optimización Cplex podremos obtener una solución optima a nuestro problema. El objetivo, es utilizando la aproximación cuadrática al problema con función de pesos exponencial (2.45), vamos aproximando a la solución de dicho problema (2.38) . El algoritmo seguirá los siguientes pasos:

1. Semilla inicial, para x^{o}=(0,0,...,0).
2. Obtenemos: $P_0 = \sum_{i=1} p_i {x^o}_i$.
3. Resolvemos el problema cuadrático (2.45).
4. Obtenemos la solución x un benéfico B* y un peso P*.
5. Si $P^* = P_0$, hemos llegado a la solución óptima, paramos.
6. En caso contrario, hacemos $x^o = x^*$, tomando como x^* nuevo punto de aproximación.
7. Volvemos al paso 2 y repetimos el proceso hasta llegar al valor óptimo.

2.6.2. Problema de cardinalidad restringida

A los problemas anteriores tenemos la opción de añadir la restricción de cardinalidad. Esta restricción, permite restringir el número de paquetes a introducir en la mochila. Esta restricción se plantearía de la siguiente forma:

$$\sum_i x_i \leq K \tag{2.46}$$

donde K es el número máximo de paquetes, que deseamos introducir para la mochila.

El modelo del problema de la mochila con capacidad variable ($KPVC$), quedaría de la siguiente forma:

$$(KPVC - ke) \equiv Max \sum_{i=1} b_i x_i - \lambda f(\sum_{i=1} p_i x_i) \tag{2.47}$$

Sujeto a:

$$\sum_{i=1} x_i \leq K \tag{2.48}$$

$$x_i \in \{0, 1\} \; \forall i \in I \tag{2.49}$$

En el siguiente capítulo se podrá ver casos prácticos para todos los modelos planteados en este capítulo.

Capítulo 3

Casos prácticos

En este capítulo obtendremos resultado a diferentes problemas obtenidos de la web de la Universidad del Estado de Florida [3]. Además nos interesará conocer el tiempo de ejecución del algoritmo que hemos realizado, como el tiempo exacto de los distintos modelos que planteamos en los capítulos anteriores. También hemos creado un problema con 29 productos nuevos con la intención de obtener una mayor información sobre tiempo de ejecución en un problema de mayores dimensiones. Se ha realizado tres archivos diferentes para poder obtener información sobre los modelos. El algoritmo heurístico especificado en el capítulo 3, está programado para poder seleccionar un número de paquetes que nos gustaría obtener. En cuanto a la resolución de los problemas, se realizará con el optimizador llamado Cplex, usando el lenguaje de programación C.

Los datos se visualizarán en la tabla 3.1, se puede ver los 8 problemas de la página de la Universidad del Estado de Florida [3], además el problema llamado P09, es el creado por el alumno para obtener mayor información sobre tiempo de ejecución. El tamaño de los problemas son los siguientes, 10, 5, 6, 7, 8, 7, 15, 23 y 29, respectivamente.

3.1. Resolución del modelo Clásico KP

En esta sección se mostrará la resolución de todos los problemas que aparecen en web, mediante el programa Cplex. El modelo lineal programado es el siguiente:

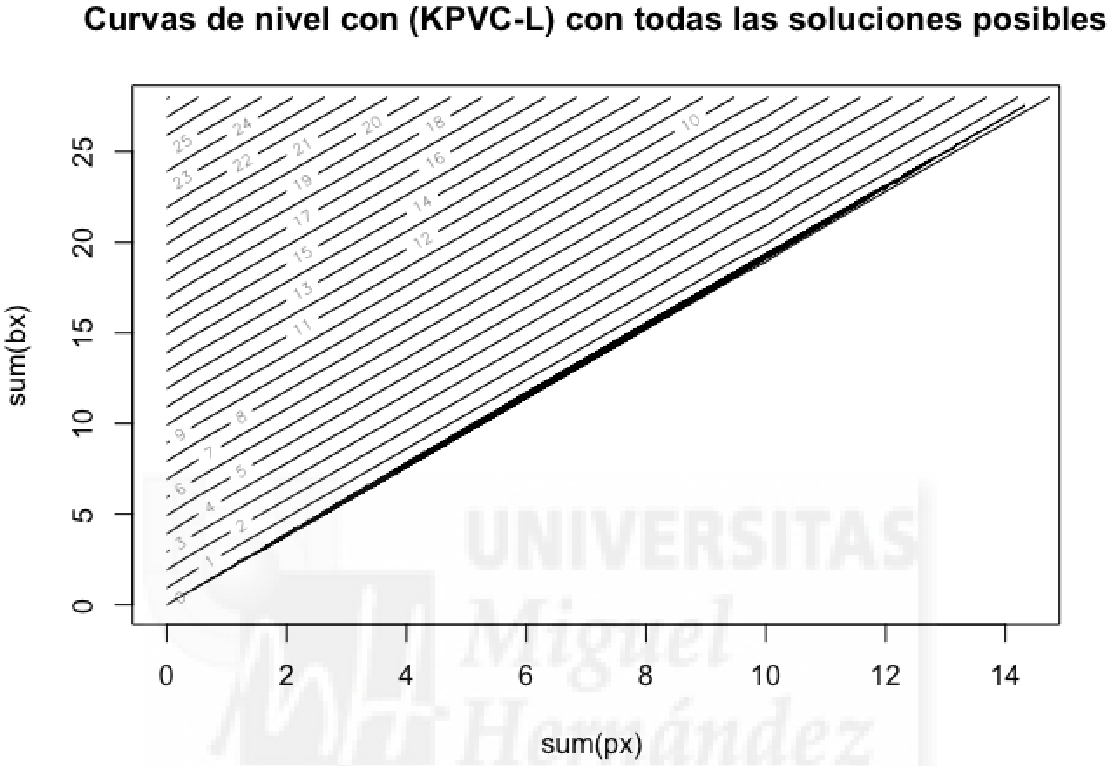
$$\begin{aligned}(KP) \equiv \quad & Max \textstyle\sum_{i=1} b_i x_i \\ & s.a. \textstyle\sum_{i=1} p_i x_i \\ & x_i \in \{0,1\} \qquad \forall i \in I\end{aligned} \tag{3.1}$$

En el modelo clásico de la mochila, pretende maximizar la suma de todos los beneficios del problema, pero con una restricción de que la suma de los pesos de los paquetes seleccionados no pueden superar una capacidad dada.

Se puede apreciar en la tabla 3.2, las soluciones del problema KP, en ella se puede ver que dimensión tiene cada problema. Además de la capacidad impuesta en los problemas, en cuanto al valor objetivo es el representado por $\sum_{i=1} b_i x_i$, como se puede ver en la formulación del problema (3.1). Las soluciones óptimas, es decir los paquetes seleccionados, coinciden con los proporcionados en la web [3]. Por último tendremos los tiempos de resolución, y vemos como no se aprecia diferencia entre el número de paquetes y el tiempo que se tarda en obtener la solución óptima.

Para representar gráficamente el modelo, tomaremos como muestra los datos del problema 7. En el gráfico se representará en el eje x el $\sum_{i=1} b_i x_i$, y en el eje y el $\sum_{i=1} a_i x_i$. Las curvas de nivel vendrán dadas por la expresión (3.2), además el λ que impondremos será de 1,9. En la diagonal se mostrará todas las soluciones posibles del problema, y la solución se obtendrá visualizando la última curva de nivel que pertenece a una solución de la diagonal.

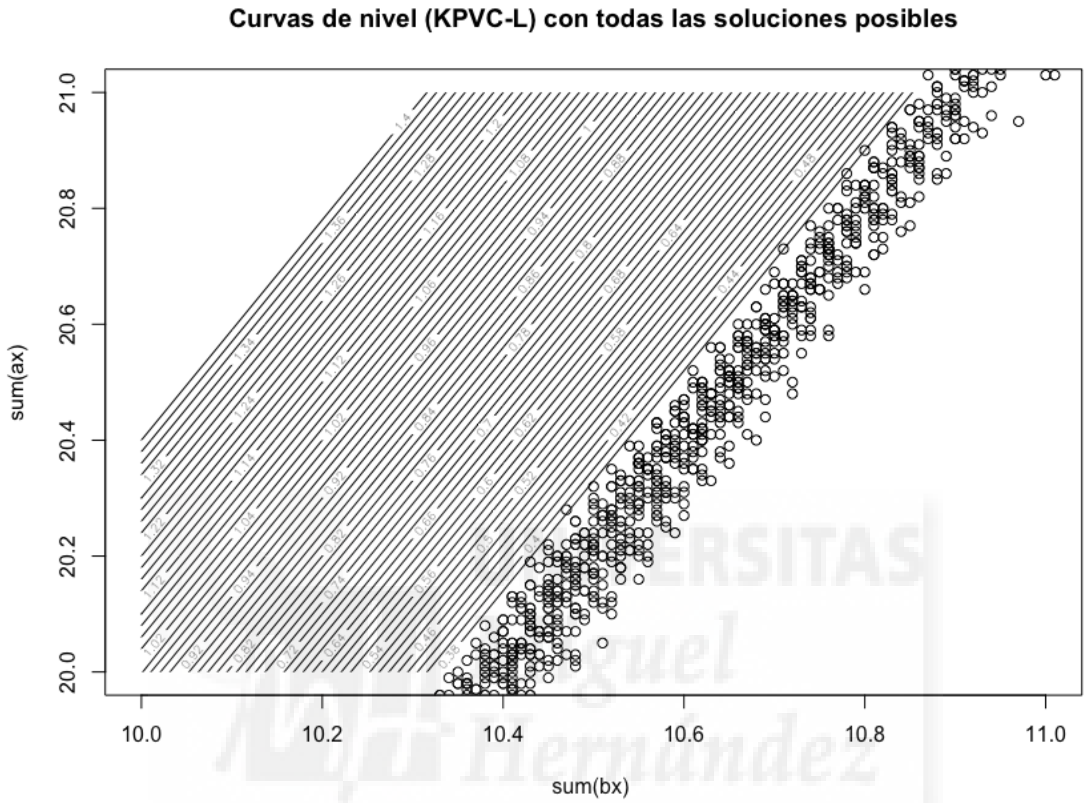
$$Max \sum_{i=1} b_i x_i - \lambda \sum_{i=1} p_i x_i \tag{3.2}$$



Al no poder visualizar correctamente la solución óptima realizaremos un zoom a la imagen. El segundo gráfico mostrado de las curvas de nivel (KPVC-l), podemos ver como la solución óptima es de 0,38, es decir obtendremos un valor del $\sum_{i=1} b_i x_i$ al rededor de 10,5 unidades y un $\sum_{i=1} a_i x_i$ al rededor de 20,3 unidades.

3.2. Resolución del modelo cuadrático

En esta sección se resolverá el modelo cuadrático con todos los datos disponibles anteriormente mostrados en las tablas. Dicho modelo tendrá la siguiente forma:



$$(KPVC - c) \equiv \quad Max \textstyle\sum_{i=1} b_i x_i - \lambda(\sum_{i=1} p_i x_i)^2 \qquad (3.3)$$
$$x_i \in \{0,1\} \qquad\qquad \forall i = 1, ..., I.$$

La capacidad mostrada anteriormente no influirá en este modelo, así que en la tabla 3.3 se puede apreciar como cambia el valor objetivo del modelo lineal al cuadrático, ya que la función de pesos se comporta de dicha forma. En lo referente al tiempo de ejecución tendremos dos tiempos a comparar, el tiempo que tarda en resolverse mediante el método exacto (es decir, se calculan todas las soluciones posibles y se resuelve la expresión $\sum_{i=1} b_i x_i - \lambda \sum_{i=1} p_i x_i$), además tenemos el tiempo de resolución de la librería Cplex. Se puede ver que la diferencia del tiempo de ejecución comienza a notarse cuando tenemos que elegir 24 paquetes del problema, si los resolvemos con Cplex solo tardaremos 0,015 segundos mientras con la resolución del procedimiento exacto tardamos 36 segundo. También hemos comprobado como el problema de 29 paquetes no ha sido capaz de resolverlo por el método exacto, a causa de la dimensión del problema. Con la ayuda de Cplex lo podemos resolver y obtenemos la solución óptima en 0,0058 segundos.

Respecto al gráfico de las curvas de nivel cuadrático, se ven como forman unas curvas convexa, dichas curvas vienen dadas por la función del modelo (KPVC-c) (3.3). Los datos seleccionados son los mismos que en la sección anterior, exceptuando el λ que se utiliza es de 0,7.

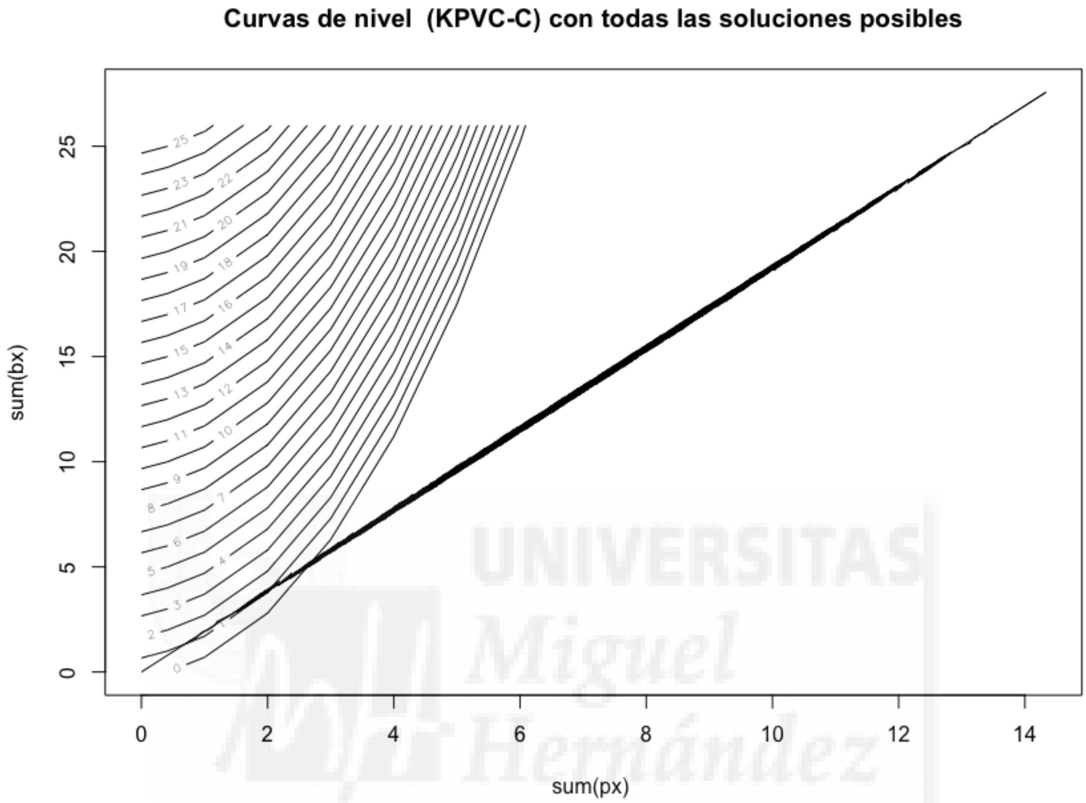
La solución óptima que se visualiza es de 1.392, se encuentra en la curva de nivel que es tangente a la diagonal de todas las soluciones posibles, esto se verá más claro en el gráfico aumentado en la región donde se encuentra dicha solución.

3.3. Resolución del modelo exponencial exacto y heurístico

El modelo a resolver en esta sección será el modelo exponencial (KPVC-e), lo que significará que la función de pesos vendrá dada por una exponencial. Es decir, el modelo exacto a resolver será el siguiente:

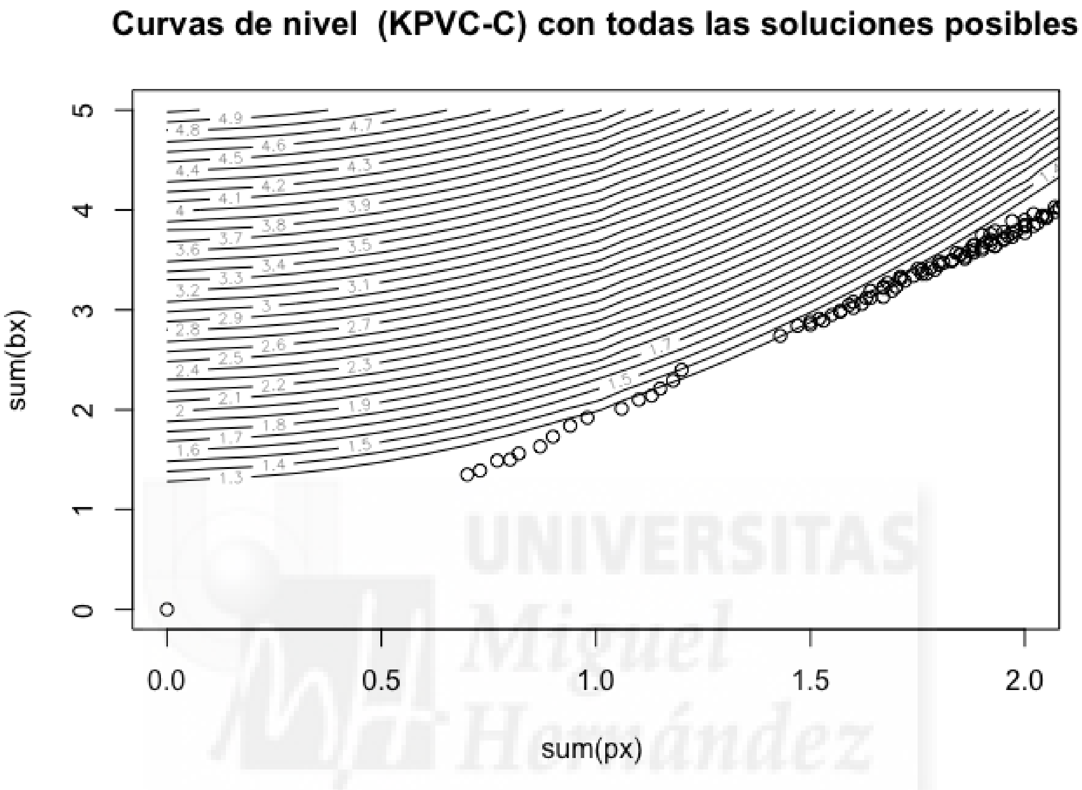
$$\begin{aligned} &Max \sum_{i=1} b_i x_i - \lambda e^{\sum_{i=1} p_i x_i} \\ &x_i \in \{0,1\}, i = 1, ..., I. \end{aligned} \tag{3.4}$$

Para resolver los problemas planteados en esta sección, se ha utilizado un scrip (mostrado en el siguiente capítulo). Primero calculamos el tiempo de ejecución del problema de manera exacta, es decir, calculando todas las posibles soluciones del



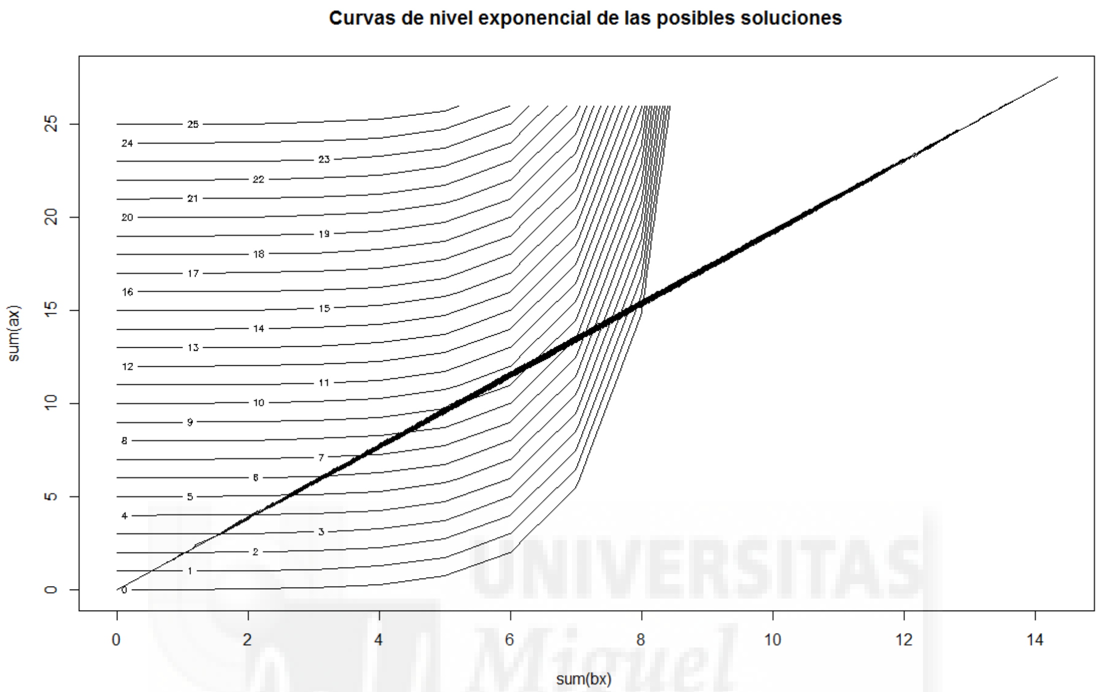
problema y obteniendo el valor máximo de dichas soluciones. Y después se ha calculado el tiempo que tarda en resolver el mismo problema con el procedimiento heurístico de aproximación. Este último se realizará por el algoritmo mostrado en la sección (2.5.1).

En cuanto al tiempo de resolución de los distintos problemas, se puede apreciar un gran cambio en el tiempo que se tarda en la resolución del problema p08 (problema con 24 paquetes). En el caso de la resolución del problema exacto, se tarda 36 segundos, en cambio con el tiempo que se tarda en obtener la solución del algoritmo es considerablemente inferior (0,036). Además seleccionamos 12 paquetes, es decir



la mitad de los paquetes a elegir. En cuanto al valor objetivo del problema es de 10,72.

El problema con 15 paquetes (p07), a elegir obtenemos una solución óptima de 0,71, seleccionamos los paquetes, (3,7,8,9,14,15), el tiempo de ejecución es de 0.017 segundos en tiempo exacto y 0,03 segundos mediante Cplex. En cuanto al valor objetivo se puede ver como en el gráfico donde se muestra las curvas de nivel y todas las posibles soluciones, aparece como solución 9,71. El λ que hemos impuesto a las curvas de nivel será de 0,005.



3.4. Resolución del modelo heurístico con capacidad variable y cardinalidad restringida

Como se ha comentado anteriormente los datos mostrados en esta sección se han resuelto a través del mismo scrip que el anterior modelo. La única novedad es la restricción de cardinalidad (3.13), en la tabla 3.5 se puede ver como los valores objetivos obtenidos son distintos a los de la sección anterior. Esto se debe a que hemos impuesto en todos los problemas, que seleccionemos un paquete menos de la solución obtenida en la sección 3.3.

$$(KPVC - ke) \equiv Max - \lambda e^P + \lambda e^P P - \frac{\lambda e^P}{2} P^2 + \tag{3.5}$$

$$+\sum_{i=1}(a_i - \lambda e^P b_i + \lambda e^P P b_i)x_i - \tag{3.6}$$

$$-\frac{\lambda e^P}{2}(V)^2 \tag{3.7}$$

Sujeto a:

$$\sum_{i=1} p_i x_i = V \tag{3.8}$$

$$\sum_{i=1} x_i \leq K \tag{3.9}$$

$$x_i \in \{0,1\}, i = 1, ..., I. \tag{3.10}$$

En cuanto al procedimiento de la obtención del valor óptimo de los problemas se realiza entre 4 y 5 iteraciones. Además destacamos la disminución del tiempo transcurrido del problema con 24 paquetes, pasamos de obtener la solución en 34 segundos a 0,007 segundos.

A continuación se mostrará los pasos a seguir para obtener la solución óptima del problema, a través del algoritmo anteriormente mencionado. Realizaremos la comprobación con cuatro paquetes para que sea más sencillo los cálculos, los datos que proporcionaremos serán los siguientes: b={0.5,1,2,4} , p={1,2,3,5} e impondremos un $\lambda = 0{,}001$. En la parte izquierda de la tabla 3.6 se mostrará todas las combinaciones posibles que se podrían dar. Además se muestran las iteraciones que se realizarán mediante la expresión de aproximación (2.45). En cada iteración se puede ver como se va cambiando la P en función de los resultados anteriores (recordemos que $P = \sum_{i=1} p_i x_i$). Comenzaremos con P=0, obtenemos que la solución óptima es de 7,428, es decir que seleccionaremos todos los paquetes. Para obtener el nuevo P sumaremos todos los pesos y obtendremos un valor de 11, es decir, nuestro P pasa a obtener un valor de 11. Volvemos a calcular a través de la expresión de aproximación el resultado óptimo, y nos aparece que debemos seleccionar todos los paquetes excepto el primero. Repetimos el procedimiento anterior hasta que en la iteración 7 encontramos el valor objetivo 4,097 que es el obtenido de forma exacta (tabla 4.8).

$$\sum_{i=1} b_i x_i - \lambda e^{\sum_{i=1} p_i x_i} \tag{3.11}$$

Esta expresión (3.11) se calcula para todas las soluciones posible y el mayor valor será el óptimo del problema, en la tabla 3.7 (Valores exactos de la función del problema $KPVC - e$), esta seleccionado en rojo.

Para concluir podemos decir que el valor obtenido mediante el algoritmo realizado con la aproximación heurística al modelo exponencial, coincide con la solución del modelo exponencial exacto. Además realizando un cambio de variable en el modelo obtenemos una mejora considerable en el tiempo de ejecución, y si añadimos la restricción de cardinalidad podemos seleccionar un número de paquetes exactos de la solución. Esto nos ayuda a conocer que paquetes nos interesaría desechar antes de la solución óptima. Todas las tablas y gráficos, realizadas en este trabajo son de elaboración propia del alumno.

3.5. Software y Hardware utilizado

Como se ha especificado antes, la librería utilizada de C++,para resolver los problemas ha sido Cplex con la versión v11.0 . En cuanto al sistema se ha realizado con el sistema operativo Linux de 64 bits, con un procesador Intel Xeon CPU E5410 de 2.33GHz y 4 núcleos, además de 7,8G de RAM total.

Tabla 3.1: Casos

Problema	**Datos**
P01	b=(0.92,0.52,0.49,0.68,0.60,0.43,0.67,0.84,0.87,0.72) p=(0.23,0.31,0.29,0.44,0.53,0.38,0.63,0.85,0.89,0.82) c=1.65
P02	b=(0.24,0.13,0.23,0.15,0.16) p=(1.20,0.70,1.10,0.80,0.90) c=0.26
P03	b=(0.50,0.50,0.64,0.46,0.50,0.05) p=(0.56,0.59,0.80,0.64,0.75,0.17) c=1,65
P04	b=(7.00,2.00,3.90,3.70,0.70,0.50,1.00) p=(3.10,1.00,2.00,1.90,0.40,0.30,0.60) c=5
P05	b=(3.50,4,00,4.50,0.20,0.70,0.08,0.05,0.05) p=(2.50,3.50,4.50,0.50,2.50,0.30,0.20,0.20) c=10.40
P06	b=(4.42,5.25,5.11,5.93,5.46,5.64,6.17) p=(0.41,0.50,0.49,0.59,0.55,0.57,0.60) c=1.70
P07	b=(1.35,1.39,1.49,1.50,1.56,1.63,1.73,1.84, 1.92,2.01,2.10,2.14,2.21,2.29,2.40) p=(0.70,0.73,0.77,0.80,0.82,0.87,0.90,0.94, 0.98,1.06,1.10,1.13,1.15,1.18,1.20) c=7.50
P08	b=(0.82,1.67,1.67,1.52,0.94,0.09,0.06,1.29,1.67,1.90,1.84,1.04, 1.25,1.31,0.95,2.06,0.67,0.85,1.82,0.06,0.90,0.57,0.46,0.36) p=(0.38,0.79,0.90,0.72,0.46,0.04,0.03,0.69,0.82,0.90,0.85,0.55, 0.61,0.67,0.48,0.95,0.32,0.44,0.93,0.03,0.49,0.26,0.22,0.16) c=6.404180
P09	b=(0.9,0.5,0.49,0.68,0.6,0.43,0.67,0.84,0.87,0.7,0.8,1.67,1.67,1.52, 0.94,0.09,0.06,1.29,1.67,1.9,1.84,1.04,1.25,1.3,0.95,2.06,0.67,0.85,1.82) p=(0.23,0.31,0.29,0.44,0.53,0.38,0.63,0.85,0.89,0.82,0.38,0.79,0.9, 0.72,0.46,0.04,0.03,0.7,0.8,0.9,0.85,0.55,0.61,0.6,0.4,0.9,0.3,0.4,0.9) c=6.404180

Nº de paquetes	Capacidad	Valor objetivo	Paquetes escogidos	Tiempo de ejecución
5	2,600	0,51	{2,3,4}	0,004955
6	1,900	1,5	{1,2,5}	0,005336
7	5,000	10,7	{1,4}	0,00495
7	1,700	17,35	{2,4,7}	0,005196
8	10,400	9	{1,3,4,5,7,8}	0,005924
10	1,650	3,09	{1,2,3,4,6}	0,005068
15	7,500	14,58	{1,3,5,7,8,9,14,15}	0,007108
24	6,404	13,549094	{1,2,4,5,6,10,11,13,16,22,23,24}	0,008588

Tabla 3.2: Modelo KP(Knapsack problem clásico) (3.1)

Nombre del problema	Nº de paquetes	Lamda	Valor objetivo	Paquetes escogidos	Tiempo Cplex	Tiempo exacto
p2	5	0,07	0,1453	{2}	0,006041	0,000056
p3	6	0,4	0,471	{1,2}	0,00468	0,000087
p4	7	0,12	8,92	{1,2,3,4}	0,004865	0,000092
p6	7	2	13,4638	{1,2,3,4,7}	0,006189	0,000089
p5	8	0,018	10,022	{1,2,3,4}	0,003936	0,000136
p1	10	0,45	1,934195	{1,2,3,4}	0,005014	0,000463
p7	15	0,7	1,392	{15}	0,006821	0,008
p8	24	0,3	3,861118	{1,6,10,16,22,24}	0,015408	36,31
p9	29	0,3	4,2955	{1,7,8,12,17,22,28}	0,005874	-

Tabla 3.3: Modelo función de peso cuadrático(KPVC-c) (3.3)

Nombre del problema	Nº de paquetes	Nº de paquetes seleccionados	Lamda	Valor objetivo exacto	Tiempo exacto	Valor obj. Heurístico	Tiempo heurístico
p02	5	{1,2,3}	0,01	0,399145	0,000043	0,399145	0.011387
p03	6	{1,2,3,4,5}	0,01	2,317809	0,000053	2,317809	0,00717
p04	7	{1,3}	0,01	9,259781	0,000064	9,259781	0,019566
p06	7	{1,2,3,4,5,7}	0,4	23,098454	0,000079	23,098454	0,011828
p05	8	{1,2}	0,001	7,096571	0,000116	7,096571	0,022827
p01	10	{1,2,3,4,5,6,7,8,9}	0,01	5,12367	0,000422	5,12367	0,007299
p07	15	{3,7,8,9,14,15}	0,005	9,712471	0,017948	9,712471	0,039272
p08	24	{1,2,4,6,10,11,13,16,17,20,22,24}	0,005	10,729855	36,007422	10,729855	0,032167
p09	29	{1,7,8,10,12,16,17,19,22,23,28}	0,005	-	-	11,189806	0,03643

Tabla 3.4: Modelo función de pesos exponencial (KPVC-e) (3.4)

Nombre del problema	Nº de Elementos	Lamda	Valor obj heurístico	Nº de artículos	Tiempo heurístico	Paquetes cogidos	*Tiempo total del proceso /s
p2	5	0,01	0,370258	2	0,00570258	{1,3}	0,00573258
p3	6	0,01	1,569713	3	0,006049	{1,2,3,4,5}	0,006089
p4	7	0,01	6,778021	1	0,00629	{1}	0,006345
p6	7	0,4	19,170229	5	0,007	{2,3,4,6,7}	0,007068
p5	8	0,001	4,409983	1	0,006279	{2}	0,006382
p1	10	0,01	4,665758	7	0,00782	{1,2,4,5,7,8,9}	0,008232
p7	15	0,005	9,554279	5	0,007706	{9,11,13,14,15}	0,024828
p8	24	0,005	8,8949	5	0,007248	{2,10,11,16,19}	35,843124
p9	29	0,005	5,725601	3	0,006966	{16,17,22}	0,006966

Tabla 3.5: Modelo de aproximación de la función de pesos exponencial con cardinalidad restringida (3.9)-(3.10)-(3.11)-(3.12)-(3.13)(3.14)(KPVC-ke)

				P* = 0	P*= 11	P*= 10	P*= 9	P*= 8	P*= 7	P*= 6
x1	**x2**	**x3**	**x4**	**Iter. 1**	**Iter. 2**	**Iter.3**	**Iter.4**	**Iter.5**	**Iter.6**	**Iter.7**
0	0	0	0	-0,001	-3023,644	-903,085	-263,350	-74,524	-20,288	-5,245
1	0	0	0	0,498	-2454,340	-715,360	-202,077	-54,648	-13,756	-2,929
0	1	0	0	0,995	-1944,910	-549,662	-148,907	-37,752	-8,321	-1,017
0	0	1	0	1,992	-1494,854	-405,490	-103,340	-23,338	-3,483	0,991
0	0	0	1	3,982	-774,364	-183,225	-36,515	-3,452	2,903	3,798
1	1	0	0	1,492	-1495,354	-405,990	-103,840	-23,838	-3,983	0,491
1	0	1	0	2,487	-1105,172	-283,844	-66,376	-12,405	-0,242	2,097
1	0	0	1	4,475	-504,430	-105,632	-15,758	1,519	**3,952**	**4,097**
0	1	1	0	2,982	-775,364	-184,225	-37,515	-4,452	1,903	2,798
0	1	0	1	4,968	-294,371	-50,066	-3,103	**3,510**	3,903	3,991
0	0	1	1	5,959	-143,685	-16,026	**1,948**	3,019	3,258	3,983
1	1	1	0	3,475	-505,430	-106,632	-16,758	0,519	2,952	3,097
1	0	1	1	6,450	-53,374	**-4,513**	-1,603	-0,952	1,017	3,071
1	1	0	1	5,459	-144,185	-16,526	1,448	2,519	2,758	3,483
0	1	1	1	6,939	**-22,937**	-15,026	-13,258	-7,905	-2,321	1,755
1	1	1	1	**7,428**	-52,374	-47,566	-33,015	-17,838	-6,756	0,037

Tabla 3.6: Iteración del algoritmo del problema KPVC-e

Tabla 3.7: Valores exactos de la función del problema KPVC-e

x1	**x2**	**x3**	**x4**	**Solución exacta**
0	0	0	0	-0,001
1	0	0	0	0,497
0	1	0	0	0,993
0	0	1	0	1,980
0	0	0	1	3,852
1	1	0	0	1,480
1	0	1	0	2,445
1	0	0	1	**4,097**
0	1	1	0	2,852
0	1	0	1	3,903
0	0	1	1	3,019
1	1	1	0	3,097
1	0	1	1	-1,603
1	1	0	1	2,519
0	1	1	1	-15,026
1	1	1	1	-52,374

Capítulo 4

Anexo

En este capítulo se plasma el código realizado para la resolución de todos los cálculos del modelo exponencial. El optimizador utilizado es el llamado Cplex y con ayuda del Notepad ++ para la elaboración del código.

En primer lugar el código nos requerirá que le introduzcamos tres ficheros de los que el primer de ellos contendrá los parámetros del problema (número de paquetes del problema, número de paquetes a seleccionar,λ,P inicial), los datos de este fichero debe estar introducido por este orden. En cuanto al segundo fichero deben ser el vector de beneficios de los paquetes, este vector debe ser introducido en columna. Por último se introducirá de la misma manera el vector de pesos que nos reporta cada paquete.

```
char *fich_TAM=argv[1];
char *fich_VECT_A=argv[2];
char *fich_VECT_B=argv[3];
FILE *param= fopen(fich_TAM,"r");
FILE *vector_a= fopen(fich_VECT_A,"r");
FILE *vector_b = fopen(fich_VECT_B,"r");
 fscanf(param,"%d",&n);
```

```
fscanf(param,"%d",&k); //numero de paquetes seleccionados
fscanf(param,"%f",&landa);
fscanf(param,"%f",&p);
a=(float *) malloc (n * sizeof(float));
for(i=0;i<n;i++){
fscanf(vector_a,"%f",&a[i]);
}
b=(float *) malloc (n *sizeof(float));
for(j=0;j<n;j++){
fscanf(vector_b,"%f",&b[j]);
}
```

Cuando tengamos todos los datos introducidos, crearemos un vector que nos calcule la parte lineal y la constante de la expresión heurística.

```
z = (float *) malloc (n *sizeof(float));
for(i=0;i<n;i++){
z[i] = a[i] - landa * exp(p) * b[i] + landa * exp(p) * p* b[i]  ;
}
alpha = (2*(landa * exp(p))/2);
cons = - landa * exp(p) + landa * exp(p) * p -((landa * exp(p)/2) * pow(p,2));
```

La siguiente parte del scrip es el cálculo de la solución exacta del modelo, es decir calculamos todas las posibles soluciones y seleccionamos la mayor.

```
if(n<25){
int iter1,band,t,cont;
t = 0;
band = 0;
int **mat = (int **)malloc(exp1 * sizeof(int *));
```

```
for (i=0; i<exp1; i++){
mat[i] = (int *)malloc(n * sizeof(int));}
iter1=exp1/2;
for(j=0;j<n;j++){
for(i=0;i<exp1;i++){
if(band==0){
mat[i][j]=0;
t++;                }
else if(band == 1){
mat[i][j] = 1;
t--;
}
if(t == iter1){
band = 1;                                   }
if(t==0){
band = 0;                }}
iter1=iter1/2;    }
float *sol_sum_a =(float *) malloc (exp1 *sizeof(float));
float q_a;
for(i=0;i<exp1;i++){
float sol_sum=0.0;
for(j=0;j<n;j++){
q_a = (mat[i][j]*a[j]);
sol_sum=sol_sum+q_a;}
sol_sum_a[i]=sol_sum;}
float *sol_sum_b =(float *) malloc (exp1 *sizeof(float));
float q_b;
```

```
for(i=0;i<exp1;i++){
float sol_sum=0.0;
for(j=0;j<n;j++){
q_b = (mat[i][j]*b[j]);
sol_sum= sol_sum+q_b;}
sol_sum_b[i]= sol_sum;}
float *solucion_op=(float *) malloc (exp1 *sizeof(float));
for(i=0;i<exp1;i++){
solucion_op[i] = sol_sum_a[i]-landa*(exp(sol_sum_b[i]));}
for(i=0;i<exp1;i++){
if(solucion_op[i]>mayor){
mayor=solucion_op[i];
posicion = i; } }
printf("\nSolucion real optima..= %f\n",mayor);
}
```

En este periodo calculamos el tiempo que tarda en realizar los cálculos También se destaca que a partir de 25 paquetes no se ha conseguido obtener resultado de dicho problema. Esta es una de las ventajas de nuestro algoritmo, es decir, no tenemos problemas al calcular soluciones para un número de paquetes superiores a 25.

El siguiente paso que se realizará será, crear un fichero con extensión .lp donde se reflejará un fichero con un formato especifico reflejado en la web de IBM [4].En el fichero estará dividido en 3 partes: la primera vendrá dada por la función objetivo, la segunda por las restricciones y la última por la declaración de las variables binarias. Como se puede ver se ha realizado dos restricciones que son las siguiente:

$$\sum_{i=1} x_i \leq k \tag{4.1}$$

$$\sum_{i=1} b_i x_i \tag{4.2}$$

```
fprintf(model,"Maximize \n");
fprintf(model,"obj: ");
for(i=0;i<n;i++){
fprintf(model," + %f x%d",z[i],i);}
fprintf(model," + [");
fprintf(model," - %f v * v",alpha);
fprintf(model,"] /2\n");
fprintf(model,"Subject To\n");
fprintf(model,"C1: ");
for(i=0;i<n;i++){
fprintf(model,"+x%d ",i);}
fprintf(model,"<= %d\n",k);
for(i=0;i<n;i++){
fprintf(model," + %f  x%d ",b[i],i);}
fprintf(model,"-v = 0\n");
fprintf(model,"BINARY\n");
for(i=0;i<n;i++){
fprintf(model," x%d ",i);}
fprintf(model,"\nEnd");
```

Cuando tenemos todo el modelo escrito en el fichero .lp crearemos punteros y obtendremos las soluciones al problema. Recordemos que la primera solución que obtengamos será con la semilla que hemos introducido al principio, que en nuestro caso será de P=0.

```
lp= CPXcreateprob (env,&status,probname);
status = CPXreadcopyprob (env,lp,"modelo.lp",NULL);
if(status){
fprintf(stderr,"Fallo al leer el problema ....\n");}
```

```
if(lp==NULL){
fprintf(stderr,"Fallo al crear el problema.\n");}
status = CPXmipopt(env,lp);
if(status){
fprintf(stderr,"Fallo en resolver el problema ....\n");
printf("\n %d error al resolver ",status);
status = CPXwriteprob(env,lp,"Problema_error.lp",NULL);}

status=CPXgetmipx(env,lp,x,0,CPXgetnumcols(env,lp)-1);
if(status) fprintf(stderr,"Fallo en obtner la  solucion  x....\n");
status=CPXgetobjval(env,lp,&objval);

while(iter <= 30){
printf("\n Iteraccion =  %d \n",iter);
p=0.0;
sum=0.0;
iter++;
for(j=0;j<n;j++) {
sum = b[j] * x[j];
p= p+sum;}
cons = - landa * exp(p) + landa * exp(p) * p -((landa * exp(p)/2) * pow(p,2));
for(i=0;i<n;i++){
z[i] = a[i] - landa * exp(p) * b[i] + landa * exp(p) * p* b[i]  ;}
for(i=0;i<n;i++){
status=CPXchgcoef(env,lp,-1,i,z[i]);}
alpha = (-(landa * exp(p))/2)*2;
status = CPXchgqpcoef(env,lp,n,n,alpha);
status = CPXwriteprob(env,lp,"Problema_error.lp",NULL); }
```

```
if(iter==2) {status=CPXwriteprob(env,lp,"problema_Iter2.lp",NULL);}
status = CPXmipopt(env,lp);
if(status){
status = CPXwriteprob(env,lp,"Problema_error.lp",NULL);}
status=CPXgetmipx(env,lp,x,0,CPXgetnumcols(env,lp)-1);
status=CPXgetobjval(env,lp,&objval1);
printf("valor objetivo = %f\n",(objval1+cons));
obj[iter]=objval1+cons;
if(obj[iter]==obj[iter-1]){break;}
}
```

Este es el algoritmo realizado, en el que se compara la solución obtenida en el paso anterior con la nueva. Entonces si coinciden es porque hemos llegado al resultado final y podemos asegurar cual es el máximo valor objetivo y cuales son los paquetes seleccionados para obtener ese resultado.

Bibliografía

[1] Hamdy A. Taha y H.D. Sherali, *Operations Research: An Introduction*, Novena edición, Pearson, México, 2012.

[2] Kellerer, Hans , Pferschy, Ulrich y Pisinger, Davi, *Knapsack Problems*, Primera edición ,Springer, New York, 2004.

[3] Universidad de Florida

http : //people.sc.fsu.edu/ jburkardt/datasets/knapsack_01/knapsack_01.html

[4] Página oficial de IBM

http : //www.ibm.com/support/knowledgecenter/es/$SSSA5P_1$2,6,0/
ilog.odms.cplex.help/CPLEX/FileFormats/topics/LP.html

[5] Eduardo Stordeur *La Eficiencia de Pareto y las Teorías Deontológicas: una respuesta libertaria a Kaplow y Shavell.*