Criterios de Parada Difusos en el Problema de la Mochila

J.L. Verdegay
Depto. de CC. de la Computación e I.A.
Universidad de Granada
E–18071 Granada
verdegay@goliat.ugr.es

E.R. Vergara
Depto. de Matemáticas
Universidad Nacional de Trujillo
Trujilllo-Perú
edmundo@goliat.ugr.es

Resumen

En este artículo se introducen criterios de parada difusos en los algoritmos del Problema de la Mochila, y como consecuencia se obtiene dos nuevos algoritmos, cuyo funcionamiento se ilustra con algunos ejemplos.

Palabras Clave: Algoritmos, Problema de la Mochila, Reglas difusas.

1 Introducción

Como es de sobra conocido, las metodologías asociadas a los conjuntos difusos se han apoyado prácticamente siempre en las que previamente existían (clásicas), siendo poco frecuente el que los modelos convencionales se traten de resolver a partir de métodos originales del campo difuso. Sin embargo esto no se ha dado con los Sistemas Basados en Reglas, que si han aportado una metodología propia proveniente de los conjuntos y los sistemas difusos. Dentro del campo de la Inteligencia Artificial, y también en otros campos, una de las áreas más provechosas, en diferentes sentidos, es la de los modelos de programación matemática, y dentro de estos, uno de los problemas más relevantes, tanto por sus aplicaciones como por ser un autentico banco de pruebas metodológicas, es el Problema de la Mochila (PM). Desde este doble punto de vista, el ámbito en el que se desarrolla este trabajo es el del interfaz entre los Sistemas Basados en Reglas y el PM.

El PM consiste en llenar una mochila de forma que se optimice el valor de los objetos transportados, respetando una limitación de capacidad. Este problema es NP-completo [3], es decir, que no se conoce un algoritmo polinomial, que obtenga su solución óptima. En estas circunstancias, el uso de los algoritmos aproximados, o heurísticas, es muy interesante porque permiten obtener soluciones que si bien no son óptimas, si están cercanas a ellas. De cara a su solución, también es una alternativa, tanto para problemas NP-completos como para el PM en particular, considerar criterios de parada difusos (basados en reglas difusas) en los algoritmos exactos conocidos.

Clásicamente los criterios de parada fijan las condiciones de finalización del procedimiento iterativo de un algoritmo, estableciéndose dichos criterios a partir de las características teóricas del problema, del tipo de solución que se busca y del tipo del algoritmo que se utiliza, que en definitiva determina un conjunto de referencia, y se detiene cuando se verifica el criterio de parada.

La flexibilización de los algoritmos exactos con la introducción de criterios de parada basados en reglas difusas, supone considerar que el conjunto de referencia es un conjunto difuso, y los criterios de parada difusos se fijan en función del grado de pertenencia de los elementos. Estos criterios de parada difusos fueron introducidos por primera vez en el ámbito de la programación matemática en [5], y luego desarrollados en [1] para el caso particular del algoritmo simplex. En [6] se han generalizado los resultados de [1] y [5] usando funciones de pertenencia no lineales, y se han estudiado criterios de parada difusos para el algoritmo de Karmarkar y sus extensiones.

Se sabe que en los algoritmos exactos del PM, los criterios de parada varían de un algoritmo a otro, pero al introducir estos criterios difusos uniformizamos la diversidad de los criterios de parada convencionales, en la siguiente sección recordamos algunos algoritmos exactos y un algoritmo para la obtención de cotas superiores del PM. En la sección 3 presentamos los criterios de parada difusos, y en la sección 4, los resultados numéricos y las conclusiones.

2 Algoritmos para el Problema de la Mochila

Formalmente planteado, el PM 0-1 consiste:

$$Max\left\{\Sigma_{j\in J}\, p_j x_j : \Sigma_{j\in J}\, w_j x_j \le c;\ x_j \in \{0,1\},\ j\in J\right\} \qquad (1)$$

siendo J={1,...,*n*} el conjunto de índices que referencia los objetos o ítems para rellenar la mochila, *c* la capacidad de la mochila, w_j, y p_j $j \in$ J el *peso* y *valor* respectivamente del j-ésimo ítem, y x_j $j \in$J, tomando valor 1 si el ítem *j* es seleccionado para introducirlo en la mochila y 0 en otro caso. Convenimos que los ítems satisfacen:

$$\frac{p_1}{w_1} \geq \frac{p_2}{w_2} \geq \ldots \geq \frac{p_n}{w_n}. \qquad (2)$$

y en lo que sigue, notaremos por *z*(PM) a la función objetivo del PM.

2.1 Cota superior de Dantzig:

Una cota superior U de (1) es tal que *z*(PM)≤U. Existen varios algoritmos para determinar diferentes cotas superiores [3]. Por su simpleza aquí utilizaremos la cota superior de Dantzig, obtenida a partir de la solución de la relajación continua (C(PM)) que es el mismo problema (1) con la condición $x_j \in [0,1]$. Conociendo el ítem crítico

$$s = \min \left\{ j : \sum_{i=1}^{j} w_i > c \right\} \qquad (3)$$

la cota de Dantzig es el mayor entero del valor óptimo z(C(PM)) de C(PM), es decir,

$$U = \lfloor z(C(PM)) \rfloor = \sum_{j=1}^{s-1} p_j + \lfloor \bar{c}\ p_s / w_s \rfloor, \qquad (4)$$

$$\bar{c} = c - \sum_{j=1}^{s-1} w_j \qquad (5)$$

donde $\lfloor x \rfloor$ nota el mayor entero del valor *x*.

2.2 Algoritmos exactos

Como comentamos, existen varios algoritmos para resolver el PM. Una muy buena recopilación de dichos algoritmos puede encontrarse en [3]. En cualquier caso aparte de los métodos "greedy" conocidos, los enfoques más relevantes son los de ramificación y acotación (Branch and Bound), y el de la Programación Dinámica. Por eso en este trabajo hemos elegido el algoritmo de Horowitz-Sahni, del tipo de ramificación y acotación, y un algoritmo basado en Programación Dinámica para ilustrar la propuesta de uso de los criterios de parada difusos.

2.2.1 El algoritmo de Horowitz-Sahni

El algoritmo de Horowitz y Sahni (HS) [2], que se destaca por su eficiencia, elegancia y simplicidad de implementación, sigue la estrategia de ramificación binaria en profundidad y se puede resumir como sigue:

En cada nodo se selecciona el ítem *j* con el máximo valor por unidad de peso de entre los ítems que aún no están seleccionados, y se generan dos nodos descendientes asignando a x_j respectivamente el valor 1 y 0. La búsqueda continúa desde el nodo asociado con la selección del ítem j (condición x_j=1). Y finalmente revisa las decisiones tomadas anteriormente mediante movimientos de retroceso. Un movimiento hacia adelante consiste en insertar el mayor número posible de nuevos ítems consecutivos en la solución actual, y un movimiento de retroceso consiste en borrar el más reciente ítem insertado de la solución actual. El Criterio de parada de este algoritmo, consiste en que el algoritmo finaliza cuando no se puede efectuar ningún retroceso.

2.2.2 El algoritmo dinámico con estados reducidos

Desde el punto de vista de la Programación Dinámica, la descomposición natural del PM consiste en los sub-problemas PM(m,c^I), con m ($1 \leq m \leq n$) ítems, y con capacidad c^I , $0 \leq c^I \leq c$. Sea

$$f_m(c^I) = max \left\{ \sum_{j=1}^{m} p_j x_j : \sum_{j=1}^{m} w_j x_j \leq c^I \right\} \qquad (6)$$
$$con \quad x_j \in \{0,1\},\ j = 1,\ldots, m.$$

el valor de la solución óptima de PM(m, c^I) .

Para resolver el PM la programación dinámica considera *n* etapas (igual al número de ítems) y calcula en cada una de ellas los valores dados por (6) usando la recursión clásica de Bellman-Dantzig siguiente:

$$f_m(c^I) = \begin{cases} f_{m-1}(c^I) \\ max\left(f_{m-1}(c^I),\ f_{m-1}(c^I - w_m) + p_m\right), \end{cases} \qquad (7)$$
$$para \quad c^I = w_m,\ldots,c$$

Las soluciones factibles asociadas a estos valores se denominan estados. La solución óptima del problema es el estado correspondiente a $f_n(c)$.

Horowitz y Sahni [2] propusieron la eliminación de aquéllos estados que no influyen en la determinación de los estados óptimos, llamados estados dominados. Dando origen así al algoritmo dinámico con estados reducidos (PD). Este algoritmo en cada etapa calcula y reserva los estados no dominados, los mismos que utiliza como valores de entrada para determinar los estados de la siguiente etapa. Como resulta evidente, el criterio de parada en este algoritmo no está netamente especificado, aunque es patente que el proceso finaliza cuando se ha obtenido el estado $f_n(c)$.

3 Criterios de parada difusos para el PM

Al igual que en muchos problemas NP-completos, en el problema de la mochila se admiten soluciones aceptables aunque no sean óptimas, para lo que pueden emplearse algoritmos heurísticos. Sin embargo aquí la alternativa que proponemos es la de flexibilizar los algoritmos exactos, introduciendo criterios de parada difusos, que finalmente proporciona una nueva clase de algoritmos heurísticos basados en reglas.

Para ilustrar el enfoque propuesto, consideramos un PM como (1), en el que para dar mayor sentido al razonamiento, suponemos que su dimensión es lo suficientemente considerable, como para que no sea posible el cálculo de la solución óptima del mismo. Es evidente que no toda solución puede ser aceptable, así como que el decisor siempre tendrá una idea imprecisa del rango de aceptabilidad sobre el valor final. Esta característica imprecisa, como resulta obvio, se representa perfectamente como un conjunto difuso, en lugar de cómo un intervalo.

Consecuentemente, asumimos que el conjunto difuso que representa la aceptabilidad propuesta por el decisor se define de la siguiente manera:

$$m(z) = \begin{cases} 0 & z < L_0, \\ f(t) & L_0 \le z \le U_0. \end{cases} \qquad (8)$$

donde $f(.)$ es una función continua no decreciente con valor entre [0,1], y L_0 y U_0 son cotas, inferior y superior respectivamente, del valor óptimo z^* de la solución óptima de PM, es decir, $L_0 \le z^* \le U_0$.

En general estas cotas tienen que estar dadas por el decisor. Pero si el problema es de gran dimensión, como aquí suponemos, lo más fácil es pensar que el decisor no sea capaz de precisarlas, por lo que puede ser una muy buena referencia como cota inferior L_0, el valor que pueda proporcionar cualquier algoritmo "greedy" aplicado al PM, y como cota superior U_0, la cota de Dantzig.

Sobre la base de las cotas y de la función de pertenencia, si el decisor considerara aceptable una solución factible cuyo valor z tenga un grado de pertenencia mayor que $\alpha \in (0,1)$ entonces el criterio de parada sería:

$$f(z) \ge \alpha \quad \text{ó} \quad z \ge f^{-1}(\alpha) \qquad (9)$$

Al introducir este criterio de parada en los algoritmos estudiados, estos se flexibilizarán de manera que se puede detener el proceso de cálculo cuando se obtenga la condición (9).

Una vez fijado el grado de aceptación α existen dos posibilidades, para el valor óptimo z^*:

a) Que sea mayor que $f^{-1}(\alpha)$, en cuyo caso se obtendrá una solución aceptable,
b) Que sea menor o igual que $f^{-1}(\alpha)$, en cuyo caso el criterio de parada difuso no tendrá sentido ya que el proceso nunca terminará, razón por la cual es necesario mantener los criterios exactos, y que el criterio difuso tenga carácter subsidiario.

Una forma de reducir la posibilidad de que se presente el caso (b) consiste en utilizar una función de pertenencia $f(.)$ cóncava, ya que entonces se cumple que

$$h^{-1}(\alpha) < f^{-1}(\alpha)$$

cuando h sea cóncava.

Una función cóncava que se puede utilizar en la definición de la función de pertenencia (8) puede ser de la siguiente forma:

$$f(t) = \sqrt[n]{\frac{t - L_0}{U_0 - L_0}} \qquad (10)$$

donde $n>1$, con lo que (9) se transforma en:

$$z \ge U_0 + (U_0 - L_0)\alpha^n \qquad (11)$$

Como ilustración, a continuación modificamos los criterios de parada de los algoritmos que introdujimos en la anterior sección.

3.1 Nuevo criterio de parada en el algoritmo HS

En el algoritmo de Horowitz-Sahni el criterio de parada consiste en detener el procedimiento cuando no existe ninguna posibilidad de retroceder. Al flexibilizar el algoritmo, además de mantener el criterio anterior incluimos la condición (9). Con este criterio de parada adicional, el algoritmo HS analizará cada vez que obtiene una nueva solución, si el valor correspondiente satisface o no a la condición (9), de ser positivo el proceso termina proporcionando la correspondiente solución aceptable.

3.2 Nuevo criterio de parada en el algoritmo PD

En este algoritmo el criterio de parada indica que el procedimiento se detiene cuando se obtiene el estado con el valor $f_n(c)$. Con este criterio, el procedimiento seguirá hasta el último estado de la última etapa, aunque la solución exacta se haya obtenido en las etapas anteriores o estados anteriores. Al introducir el criterio de parada difuso, en cada etapa y en cada estado, se analizará si se ha obtenido un valor aceptable o no; si es afirmativo, se detiene el proceso. Naturalmente, con este criterio no hay necesidad de resolver todas las etapas ni todos los estados.

4 Resultados experimentales

En esta sección presentamos los resultados de los experimentos computacionales realizados con problemas de la mochila de 1.000, 5.000, 10.000 y 50.000 ítems, cuyos valores y pesos comprendidos en el intervalo [1,1000], han sido generados aleatoriamente, y se ha utilizado la semi-suma de los pesos como capacidad de la mochila.

En cada uno de los algoritmos con criterios de parada difusos vistos en la sección anterior se considera un grado de pertenencia α=0.5, α=0.8 y α=1; este último caso, como es obvio, corresponde a la solución obtenida mediante el algoritmo clásico, sin el criterio de parada

difuso. En la tabla de los resultados se consignan el error cometido y el tiempo de ejecución para cada uno de los algoritmos de la sección anterior. El error se obtiene mediante la siguiente fórmula:

$$error = \frac{(Cota - Dantzig) - (sol.obtenida)}{cota\,de\,Dantzig} * 100$$

Por otro lado, para comparar los resultados que se han obtenido, dado el carácter aproximado de los algoritmos que hemos obtenido, recurrimos al algoritmo de Sahni [4], un algoritmo muy utilizado y bien conocido, que se corresponde con el siguiente esquema "greedy": El algoritmo trata de llenar la capacidad restante de la mochila, una vez introducido un conjunto M de k ítems. Para ello, en primer lugar, se generan todos los subconjuntos M posibles de k elementos para introducirlos en la mochila. Posteriormente rellena el espacio restante de la mochila. Por cuestiones obvias, el Algoritmo de Sahni suele usarse con k = 1, 2, 3.

En los criterios de parada difusos de los dos algoritmos revisados en la sección 2, cuyos resultados presentamos en esta sección, se ha utilizado la condición:

$$z \geq L_0 + (U_0 - L_0).\alpha^4$$

es decir, los criterios difusos están definidos mediante la función de pertenencia (8) con f(.) dada por (10) y n=4; L_0 es la solución respectiva obtenida mediante el algoritmo básico greedy y U_0 es la cota de Dantzig.

La tabla 1 muestra los resultados de los experimentos, tras haber utilizado los algoritmos con criterios de parada difusos de la sección 3, y paralelamente se consignan los resultados que proporciona el algoritmo aproximado de Sahni.

Como podemos observar en la tabla 1, el algoritmo de Horowitz y Sahni con criterio de parada difuso proporciona resultados con muy pocos errores y en tiempos también muy cortos, con respecto tanto a la solución exacta como a soluciones aproximadas dadas por el algoritmo de Sahni; esta última ventaja se hace más significativa cuando n (número de ítems) se incrementa. Por otro lado, se observa que el algoritmo dinámico con criterios de parada difusos proporciona una mejor aproximación (menor error) y en menor tiempo que el algoritmo de Sahni lo que junto con los resultados anteriores validan el enfoque propuesto.

5 Conclusiones

A la vista de todo lo anterior podemos concluir que la incorporación, a los algoritmos convencionales aquí considerados, de criterios de parada basados en reglas difusas, produce nuevos procedimientos de solución que, en las situaciones en que es difícil la obtención de soluciones óptimas, permiten resolver los problemas en cuestión a satisfacción del decisor. La generalidad y robustez del método que se deriva del uso de los sistemas basados en reglas, mas que contrastado en muchos contextos, hacen prever la adecuación del enfoque aquí descrito a una gran variedad de situaciones.

Referencias

[1] F. Herrera y J.L. Verdegay. Fuzzy control rules in optimization problems, *Scientia Iranica*, Vol 3, Nos.1,2,3. (1996) 89-96.

[2] E. Horowitz y S. Sahni. Computing partitions with applications to the knapsack problem, *Journal of ACM* 21 (1974) 277-292.

[3] S. Martello y P. Toth. *Knapsack Problems*, John Wiley and Sons 1990.

[4] S. Sahni. Approximate algorithms for the 0-1 knapsack problem, *Journal of ACM* 22 (1975) 115-124.

[5] J.L. Verdegay. Fuzzy Optimization: models, methods and perspectives; *6th IFSA-95 World Congress*. Sau Paulo - Brazil, 1995, pp. 39-7.

[6] E.R. Vergara. *Nuevos Criterios de Parada en Algoritmos de Optimización*, Dpto. de CC. de la Computación e I.A. (Tesis Doctoral), 1999.

Tabla 1. Resultados de los experimentos, mediante HS y PD con criterios de parada difusos.

n	α=1 (exacto)				α=0.5				α=0.8				Algoritmo de Sahni			
	HS		PD		HS		PD		HS		PD		k=2		k=3	
	Error	tiemp	error	tiemp	error	tiemp	Error	tiemp	Error	tiemp	Error	Tiemp	error	Tiemp	Error	Tiemp
1.000	.00192	0.044	.00192	0.24	.00517	0.000	.00436	0.19	.00395	0.000	.00382	0.23	.00325	0.11	.00247	0.23
5.000	.00009	1.462	.00009	3.77	.00046	0.000	.00027	3.07	.00031	0.022	.00024	3.10	.00023	2.54	.00017	4.58
10.000	.00002	3.720	.00002	12.19	.00011	0.010	.00006	10.18	.00006	0.012	.00003	10.25	.00029	8.80	.00027	13.97
50.000	.00000	21.838	.00000	223.94	.00002	0.096	$4/10^7$	218.35	.00001	0.136	$3/10^7$	218.37	$3/10^7$	212.48	.00000	257.76