Búsqueda multiobjetivo basada en RBFS y Punto Ideal

Javier Coego, Lorenzo Mandow y José Luis Pérez de la Cruz*

Universidad de Málaga, Andalucía Tech. Departamento de Lenguajes y Ciencias de la Computación.{jcoego, lawrence, perez}@lcc.uma.es

Resumen El algoritmo *RBFS* es una alternativa eficiente para la búsqueda heurística con retroceso del camino mínimo en árboles con un consumo lineal de memoria. Este artículo presenta $RIPS$, un algoritmo multiobjetivo exacto, basado en *RBFS* y en el concepto de *punto ideal*, que localiza el conjunto de soluciones óptimas de Pareto. Se presentan los resultados de una evaluación empírica entre este algoritmo y una adaptación anterior de *RBFS* al caso multiobjetivo. Los resultados muestran que $RIPS$ supone una mejora sobre dicho algoritmo.

1. Introducción

La búsqueda del camino óptimo en un grafo es un problema fundamental en inteligencia artificial. En muchos problemas, la elección del camino óptimo involucra a más de un objetivo, cada uno con su propia función de coste. Las soluciones óptimas son los denominados *óptimos de Pareto*, en los que no es posible mejorar un objetivo sin empeorar al menos otro. En búsqueda en grafos el número de soluciones crece en general exponencialmente incluso para el caso de dos objetivos.

Dentro de la búsqueda multiobjetivo exacta, la búsqueda *depth-first* con profundización iterativa ha sido ampliamente tratada en diferentes trabajos [6] [1] [2]. Si bien el algoritmo *RBFS* (*Recursive Best-First Search*)[7] ha mostrado un buen comportamiento frente a las variantes de profundización iterativa en la búsqueda con un objetivo [9], esta opción apenas ha sido explorada para el caso multiobjetivo. *RBFS* expande siempre los nodos por primera vez en un orden primero el mejor. *RBFS* mantiene en memoria únicamente la rama que está explorando en cada momento, eliminando de memoria el resto de ramas visitadas y recordando el mejor coste c alcanzado por cualquiera de esas ramas *olvidadas*. Si la rama actual supera dicho coste c, se reconsidera la mejor rama *olvidada*.

Una extensión de *RBFS* al caso multiobjetivo debe contemplar, entre otras cosas, que en las ramas discontinuadas puede haber distintos vectores óptimos de Pareto. Que tengamos constancia, la única generalización de *RBFS* al caso multiobjetivo es el algoritmo, $MOMA^*0$ [4]. Este algoritmo impone un orden lexicográfico en la exploración del grafo, lo que le permite utilizar un único

* La presentación de este trabajo está subvencionada por Plan Propio de Investigación de la Universidad de Málaga - Campus de Excelencia Internacional Andalucía Tech.

vector como cota de profundización. Esta estrategia, no obstante, puede necesitar de un gran número de profundizaciones para encontrar la solución. Algunas aplicaciones de este algoritmo en el diseño de circuitos son el *scheduling* de operadores, o el encaminamiento de canales [4]. Otro problema típico es el árbol de recubrimiento mínimo multiobjetivo.

Este artículo presenta $RIPS$ (*Recursive Ideal Point Search*), una generalización alternativa de *RBFS* al caso multiobjetivo. RIPS utiliza el concepto de *punto ideal* para el cálculo de las cotas de profundización, sacrificando parcialmente el carácter *primero el mejor* de *RBFS* en aras de una mayor eficiencia. El uso del punto ideal permite considerar simultáneamente todos los objetivos, considerando un menor número de cotas y reduciéndolas a un único vector.

La estructura de este artículo es la siguiente. En la sección 2 se ilustran los conceptos principales de la búsqueda *RBFS* escalar y multiobjetivo. A continuación se describe el nuevo algoritmo $RIPS$ y se presenta un ejemplo. La sección 4 analiza el rendimiento de $RIPS$ frente a $MOMA^*0$. Por último, se destacan las conclusiones de este trabajo y posibles líneas de trabajo futuro.

2. Antecedentes

2.1. RBFS

Dado un árbol, posiblemente infinito, con raíz en s, y un conjunto de nodos objetivo Γ, consideremos el problema de buscar un camino óptimo desde s hasta cualquier $\gamma \in \Gamma$. El coste de un camino $P = (s = n_0, n_1, \ldots, n_k)$ es la suma de los costes positivos de sus arcos $g(P) = \sum_{i=0}^{k-1} c(n_i, n_{i+1})$, y disponemos de un heurístico (o cota inferior) $\forall n \; h(n) \geq 0$ del coste desde cada nodo n del árbol hasta un nodo de Γ. La *evaluación heurística* del coste de un camino $P_{sn} = (s, \ldots, n)$ vendrá dada por la función $f(P_{sn}) = g(P_{sn}) + h(n)$.

RBFS [7] resuelve el problema mediante una búsqueda tipo *backtrack* con profundizaciones progresivas, garantizando que los nodos se expanden por primera vez en orden *primero el mejor* según f, sea ésta monótona o no. Se define *nodo abierto* como aquel ya visitado alguna vez por el algoritmo, pero aún no expandido. *RBFS* mantiene en memoria únicamente los nodos del camino actualmente explorado, así como sus hermanos. La implementación es recursiva[1], con parámetros n (nodo a expandir), $F(n)$ (evaluación almacenada de n), y B (cota de profundización local). La llamada inicial es $RBFS(s, h(s), \infty)$. Resumimos a continuación las características principales de las operaciones *avance* y *retroceso*:

1. En su operación de *avance*, RBFS expande todos los nodos n' en el subárbol bajo n tales que su valor $f(n')$ es menor o igual que la cota superior local, i.e. $f(n') \leq B$. Aquellos nodos con $f(n') > B$ quedan abiertos y definen la frontera de búsqueda bajo n. El valor B corresponderá siempre al menor valor f de todos los nodos que quedaron abiertos en el *resto* del árbol (ramas que *no* pasan por n), garantizando así el orden de expansión *primero el mejor*.

[1] Véase [5] para una versión iterativa equivalente.

2. Una vez completada la exploración bajo el nodo n, el algoritmo *retrocederá* hasta aquel nodo bajo el que se encuentre en ese momento el nodo abierto con menor valor de f. Sea mfa(n) el menor valor de f de los nodos que quedaron abiertos bajo n. El retroceso de $RBFS(n, F(n), B)$ devuelve a su antecesor el valor mfa(n). Esto permitirá calcular las cotas de profundización en los siguientes avances. Si no hubiera sucesores, devuelve un valor infinito.
3. Destacamos por último una operación propia del *avance sobre ramas previamente exploradas*. Cuando el algoritmo expande un nodo n, almacena para cada sucesor inmediato n_i una evaluación F_i. Ésta coincidirá inicialmente con $f(n_i)$, pero se actualiza con los valores mfa(n_i) cada vez que se realiza una llamada recursiva sobre n_i. Cada llamada sobre n_i hace crecer F_i de manera estricta. Cuando en una llamada $RBFS(n, F(n), B)$ se cumple que $f(n) < F(n)$ es porque el nodo n ya fue explorado previamente. En tal caso los sucesores n_i de n pueden tomar como valor inicial $F_i = max(F(n), f(n_i))$, reduciendo el número de reexpansiones que debe realizar el algoritmo.

La búsqueda continúa hasta que: (a) se realiza una llamada sobre un nodo objetivo, devolviendo el camino encontrado hasta él; (b) se produce un retroceso desde s, terminando entonces con fracaso.

2.2. Búsqueda multiobjetivo

Consideremos ahora el caso en el que los arcos están etiquetados con un vector de coste q-dimensional $\boldsymbol{c}(n, n') = (c_1, \ldots, c_q)$, donde $c_i > 0$ representa el coste asociado al i-ésimo objetivo. El coste asociado a un camino $\boldsymbol{g}(P)$, los valores heurísticos $\boldsymbol{h}(n)$, y las evaluaciones $\boldsymbol{f}(P) = \boldsymbol{g}(P) + \boldsymbol{h}(n)$ serán de tipo vectorial.

Dados dos vectores $\boldsymbol{v}, \boldsymbol{v'} \in \mathbb{R}^q$, la relación de orden parcial $\prec$ (denotada como relación de **dominancia**) se define como $\boldsymbol{v} \prec \boldsymbol{v'} \Leftrightarrow \forall i (1 \leq i \leq q), (v_i \leq v'_i) \wedge (\boldsymbol{v} \neq \boldsymbol{v'})$, donde v_i denota el i-ésimo elemento del vector $\boldsymbol{v}$. Se define la relación de orden parcial $\sim$ (denotada como relación de *indiferencia*) como $\boldsymbol{v} \sim \boldsymbol{v'} \Leftrightarrow (\boldsymbol{v} \not\prec \boldsymbol{v'}) \wedge (\boldsymbol{v'} \not\prec \boldsymbol{v})$. Dado un conjunto de vectores X, se definen los *óptimos de Pareto* o *frontera de Pareto* de dicho conjunto, $nd(X)$, como el conjunto de vectores no dominados de X, es decir, $nd(X) = \{\boldsymbol{x} \in X \mid \nexists \boldsymbol{y} \in X \quad \boldsymbol{y} \prec \boldsymbol{x}\}$.

El *problema de búsqueda multiobjetivo* consiste en encontrar el conjunto de todos los caminos desde s a nodos en Γ con costes no dominados. El conjunto de tales costes no dominados se denota por C^*.

Las siguientes definiciones serán útiles más adelante. Dados dos vectores $\boldsymbol{v}, \boldsymbol{v'} \in \mathbb{R}^q$ decimos que $\boldsymbol{v}$ es *mejor lexicográfico* que $\boldsymbol{v'}$ ($\boldsymbol{v} \prec_L \boldsymbol{v'}$) sii $(\exists i \; v_i < v'_i) \wedge (\forall j < i \; v_j = v'_j)$. Dado un conjunto de vectores, el mejor lexicográfico es trivialmente un óptimo de Pareto. Nótese que $\prec_L$ define un orden total. Asimismo, se dice que v es *mejor lineal* que v' ($\boldsymbol{v} \prec_{lin} \boldsymbol{v'}$) sii $\sum_{i=1}^{q} v_i < \sum_{i=1}^{q} v'_i$

Dado un conjunto de vectores X, el punto ideal de X ($\boldsymbol{IP}(X)$) es un vector formado por los valores más pequeños que pueden ser localizados en cualquier vector del conjunto X para cada componente, es decir, $\forall j, 1 \leq j \leq q, \boldsymbol{IP}(X)_j =$

mín$\{\boldsymbol{v}_j \mid \boldsymbol{v} \in X\}$. Definimos la relación *estrictamente mejor* ($\ll$) como $\boldsymbol{g} \ll \boldsymbol{g'} \Leftrightarrow \forall j(1 \leq j \leq q), \boldsymbol{g}_j < \boldsymbol{g'}_j$.

2.3. MOMA*0

$MOMA^*0$ [4] es una generalización de *RBFS* al problema multiobjetivo descrito en la sección 2.2. Por simplicidad, en este trabajo consideraremos una versión de $MOMA^*0$ restringida a un único vector heurístico por nodo.

En la búsqueda multiobjetivo no existe un único valor mínimo entre todos los nodos abiertos. En general, existe un conjunto de vectores Pareto-óptimos. La generalización más directa de *RBFS* pasaría por sustituir los valores $F(n)$ y B por conjuntos de vectores Pareto-óptimos. Esto aportaría la ventaja de que el algoritmo profundizase *simultáneamente* en la búsqueda de *todas* las soluciones óptimas, resolviendo el problema en un número mínimo de profundizaciones. Sin embargo, esta estrategia acarrearía también una sobrecarga computacional muy elevada. La comprobación para discontinuar la búsqueda en una rama ya no implicaría simplemente una comparación escalar con la cota, sino comprobaciones de dominancia contra un conjunto de vectores. El cálculo y almacenamiento de estos conjuntos es también en sí mismo un desafío, ya que su tamaño puede crecer exponencialmente con la profundidad de la búsqueda en el peor caso.

Los algoritmos multiobjetivo primero el mejor solucionan en parte este problema usando un orden total en el conjunto de nodos abiertos que garantice que el mejor elemento es siempre un óptimo de Pareto. $MOMA^*0$ adopta esta estrategia, explorando las ramas del árbol según una estrategia primero el mejor en base a un orden lexicográfico. En este caso $\boldsymbol{F}(n)$ y $\boldsymbol{B}$ serán vectores[2]. Otra alternativa sería emplear un orden lineal.

Otra diferencia con *RBFS* es que $MOMA^*0$ no termina tras encontrar la primera solución, sino que continúa la búsqueda hasta encontrar todas las soluciones óptimas de Pareto. Para ello guarda en un conjunto $COSTS$ los costes de todas las soluciones encontradas. Este conjunto actúa como una cota superior global de la búsqueda, de modo que si en alguna llamada $\boldsymbol{F}(n)$ está dominado por algún vector de $COSTS$, se fuerza un retroceso devolviendo un vector de costes infinitos. La llamada inicial es $MOMA^*0(s, \boldsymbol{h}(s), \boldsymbol{\infty}, \emptyset)$. En relación a lo expuesto en la sección 2 para *RBFS*, las principales operaciones de $MOMA^*0$ quedan así:

1. Al realizar un *avance*, $MOMA^*0$ expande todos los nodos n' en el subárbol bajo n tales que $\boldsymbol{f}(n') \preceq_L \boldsymbol{B}$ y no están dominados por $COSTS$. La cota $\boldsymbol{B}$ será el mejor $\boldsymbol{f}$ lexicográfico entre los nodos abiertos del resto del árbol.
2. El *retroceso* de $MOMA^*0(n, \boldsymbol{F}(n), \boldsymbol{B}, COSTS)$ devuelve en este caso $\mathbf{mfa}(n)$, definido como el mejor lexicográfico de los nodos abiertos bajo n, y los costes de las soluciones encontradas. Si n no tiene sucesores, o todos están dominados por $COSTS$ devuelve un vector con valores infinitos y $\emptyset$.

[2] En [4] se utiliza una terminología diferente. No obstante, utilizaremos estos nombres para proporcionar una correspondencia más clara con la notación estándar de RBFS.

3. La propagación de valores cuando se avanza sobre ramas previamente exploradas se realiza mediante una función denominada *Minf*. Los detalles pueden consultarse en [4].

3. Algoritmo RIPS

En esta sección se describe $RIPS$, una nueva generalización $RBFS$ al caso multiobjetivo. Como ya se comentó en la sección 2.3, usar una cota de profundización vectorial en $MOMA^*0$ evita tests de dominancia contra la cota, que resultarían computacionalmente muy costosos. Sin embargo, el orden lexicográfico empleado por $MOMA^*0$ prioriza los objetivos de modo que no se profundizará en la consecución del primer objetivo hasta que se hayan probado (profundizado en) todos los valores del segundo, y así sucesivamente. En última instancia, esto puede propiciar un progreso muy lento del algoritmo, ya que si disponemos de q objetivos y costes óptimos *enteros* acotados por d, el número de cotas de profundización distintas puede ser $\mathcal{O}(d^q)$ en el peor caso.

$RIPS$ utiliza una cota de profundización que supone un compromiso entre la información proporcionada por *toda* la frontera de Pareto de los nodos abiertos, y la eficiencia de usar un único vector de cota (como el óptimo lexicográfico). Concretamente, $RIPS$ utiliza una cota basada en el *punto ideal* de la frontera de Pareto de los nodos abiertos [2]. Al usar un único vector como cota, se mantiene la eficiencia computacional en las comparaciones. Sin embargo, el punto ideal condensa en un sólo vector información de *toda* la frontera, ya que refleja el mejor coste alcanzado para cada uno de los objetivos durante la profundización en cualquiera de las ramas.

Al igual que en $MOMA^*0$, el algoritmo $RIPS$ utiliza un conjunto $COSTS$ como cota superior global de la búsqueda. El pseudocódigo de $RIPS$ se muestra en la tabla 1. La llamada inicial es $RIPS(s, \boldsymbol{h}(s), \boldsymbol{\infty}, \emptyset)$. Las principales operaciones pueden resumirse así:

1. Al realizar un *avance*, $RIPS$ expande todos los nodos n' en el subárbol bajo n tales que $\boldsymbol{f}(n') \ll \boldsymbol{B}$. En este caso, $\boldsymbol{B}$ corresponde al punto ideal de los vectores de coste no dominados entre los nodos abiertos del resto del árbol.
2. El *retroceso* de $RIPS(n, \boldsymbol{F}(n), \boldsymbol{B}, COSTS)$ devuelve el punto ideal de los costes de nodos abiertos bajo n, y los costes de las soluciones encontradas. Si n no tiene sucesores, o todos están dominados por $COSTS$, devuelve un vector con valores infinitos y $\emptyset$.
3. La propagación de valores cuando se avanza sobre ramas previamente exploradas se realiza a través de la función *MaxVector*, análoga a la función *Minf* usada por $MOMA^*0$ (ver [3] para más detalles de *MaxVector*).

En general, no es posible emplear el punto ideal como cota y discontinuar la búsqueda cuando los sucesores estén *dominados* por ella. Considérese un nodo n con cota de profundización $\boldsymbol{B} = (1, 1)$ y dos sucesores con valores $\boldsymbol{f}(n_1) = (2, 1)$ y $\boldsymbol{f}(n_2) = (1, 2)$. Supongamos que la búsqueda continúa hasta n_1 y se discontinúa al ser $\boldsymbol{B} \prec \boldsymbol{f}(n_1)$. La búsqueda continúa por n_2, discontinuándose igualmente

```
RIPS-base ()
      COSTS ← ∅
      (FA, COSTS) ← RIPS (s, nodomset(H(s)), ∞, COSTS)
      return (COSTS)


RIPS (n, FA_n, c_sup, COSTS)

N_Sol ← Vectores de F(n) no dominados por soluciones
SI (N_Sol = ∅)
      devolver (∞, COSTS)
N_csup ← Vectores de N_Sol no estrictamente peores que c_sup
SI (N_csup = ∅)
      devolver (N_Sol, COSTS)
SI (n es un nodo solución)
      devolver (∞, nodomset (COSTS ∪ {f(n)}))
SI (n no tiene sucesores)
      devolver (∞, COSTS)
Para cada sucesor n_i de n añadir elemento en FA con los valores
      -nodomset(F(n_i)), si n no ha sido expandido previamente
      -nodomset(MaxVector (f,FA(n), ∀f ∈ F(n_i))), en caso contrario
Disc ← ∅
LOOP
      SI (no quedan sucesores por procesar en FA)
            SI (Disc = ∅)
                  devolver (∞, COSTS)
            SINO
                  Disc_nds ← Vectores de Disc no dominados por soluciones
                  devolver (IP(Disc_nds), COSTS)
      fa ← Escoger primer elemento de FA
      N1 ← Vectores de fa no dominados por soluciones
                  y no estrictamente peores que la cota
      SI (N1 = ∅)
            Eliminar fa de FA
            Disc ← Disc ∪ Vectores de fa no dominados por soluciones
      SINO SI (en N1 hay vectores de la frontera de Pareto de FA)
            ip_csup ← IP ({c_sup ∪ (FA − fa))
            SI (hay algún vector en N1 no estrictamente peor que ip_csup)
                  (FA[1], COSTS) ← RIPS (n_1, N1, ip_csup, COSTS)
            SINO
                  Mover fa al final de FA
END LOOP
```

Cuadro 1: Algoritmo $RIPS$.

al ser $\boldsymbol{B} \prec \boldsymbol{f}(n_2)$. La nueva cota de n se calcularía como $\boldsymbol{IP}(\{(2,1),(1,2)\}) = (1,1)$, produciéndose un bucle infinito. Por este motivo $RIPS$ utiliza la relación *estrictamente mejor* para discontinuar la búsqueda, sacrificando así el carácter *primero el mejor* del algoritmo. No obstante, esto garantiza que si disponemos de costes óptimos *enteros* acotados por d, el número de cotas de profundización distintas será $\mathcal{O}(d)$ en el peor caso.

Un detalle importante en $RIPS$ es que la lista FA que contiene los sucesores no está ordenada. Cada elemento de la lista contiene los vectores de coste de un sucesor del nodo actual, y se actualiza con la expansión de los nodos bajo dicho sucesor. Siempre se procesa el primer elemento de la lista. Un nodo se elimina de FA cuando sus costes están dominados por soluciones ya encontradas o son *estrictamente peores* que la cota superior. En este segundo caso se usan estos vectores para el cálculo del punto ideal que servirá como nuevo vector de coste del padre n, al referenciar costes de caminos discontinuados, pero no descartados definitivamente. Si un sucesor es *estrictamente peor* que otros, se pasa al final de la lista FA, dando preferencia a los estrictamente mejores. En otro caso, el nodo se expande usando por un lado su estimación almacenada, y como cota superior el punto ideal de los vectores de coste de los caminos pendientes. Si la lista FA está vacía, se devuelve al nodo n información de los caminos discontinuados de interés en reexpansiones futuras. Esta información se almacena en el conjunto *Discontinued*. $RIPS$ calcula el punto ideal de dicho conjunto, el cual será el nuevo coste de n. Si no hubiera nodos en esta situación, porque fueran soluciones ya encontradas o porque su coste exceda el de las mismas, entonces n recibe como actualización de su coste el vector $\boldsymbol{\infty}$ para descartarlo definitivamente.

Para mayores detalles sobre el algoritmo $RIPS$ y demostraciones formales de su completitud y admisibilidad, puede consultarse [3].

3.1. Ejemplo

Aplicaremos $RIPS$ al árbol de la figura 1a con $\Gamma = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$. En la figura 1b se muestra la expansión del nodo s, desempatando de izquierda a derecha. FA_s toma como valor inicial $\boldsymbol{f}(s)$, y como cota de profundización el vector $\boldsymbol{\infty}$, al ser el nodo raíz. Dado que sus tres sucesores son no dominados entre sí, y suponiendo desempate de izquierda a derecha, se procesa en primer lugar n_1 (figura 1c). Al no haber expansiones previas de n_1, $FA_{n_1} = \boldsymbol{f}(n_1)$ y su cota de profundización es el *punto ideal* de los vectores de coste de los caminos a través de sus hermanos n_2 y n_3, el vector $(2,1)$. Se explorará el subárbol bajo n_1 hasta que todas las ramas exploradas sean estrictamente peores que dicha cota.

Al expandir n_1 (figura 1c), se generan los nodos γ_1 y γ_2. Al ser ambos estrictamente peores que la cota, la búsqueda se discontinúa. El punto ideal de sus costes $(3,6)$ y $(5,5)$ será el nuevo vector de coste de n_1. Este vector, $(3,5)$, es estrictamente peor que el de n_2 y n_3, con lo cual se aplaza el procesamiento de n_1. Al procesar n_2, su cota superior se calcula de modo análogo a la de n_1, resultando en el vector $(3,1)$. La expansión de n_2 (figura 1d) genera γ_3, con $(3,8)$. Se expande γ_3, y como es un nodo solución (aunque subóptimo), se añade al conjunto $COSTS$.

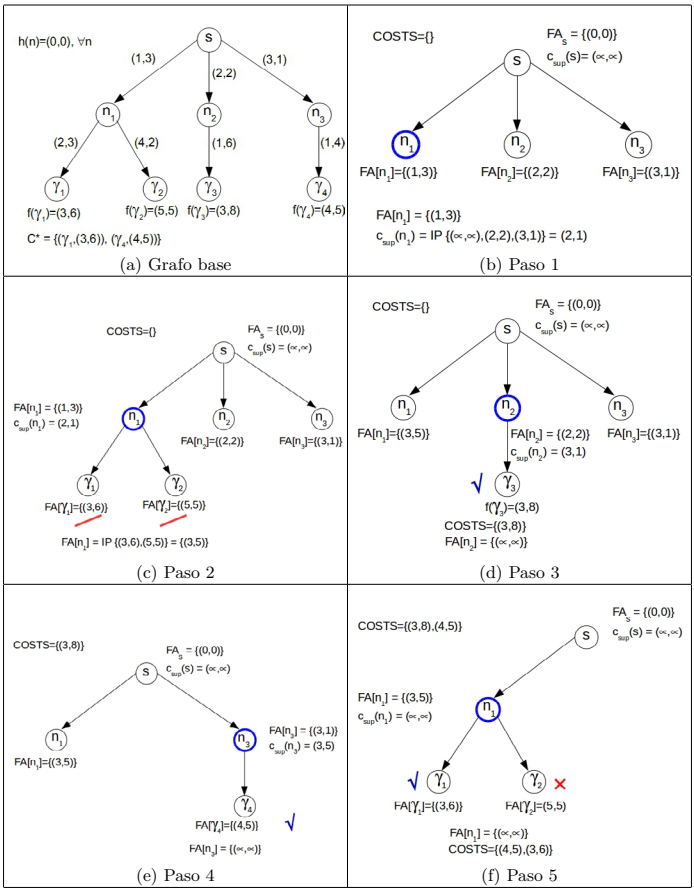


Figura 1: Resolución mediante *RIPS*.

Al no haber ramas pendientes, la estimación de n_2 se actualiza a ∞. A continuación se expande n_3 (figura 1e), calculándose su cota atendiendo a n_1. Se expande su único sucesor, γ_4, y al ser solución se añade a $COSTS$. Dado que los costes de las dos soluciones encontradas son no dominados entre sí, ambas soluciones se mantienen. Al no haber ramas pendientes a través de n_3, su estimación será ∞. Al estar dominado por $COSTS$, se descarta.

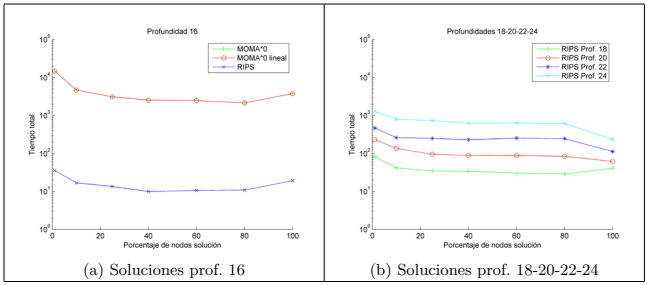
La única rama pendiente es explorar n_1, con cota ∞ (figura 1f). La rama n_1 ya fue explorada, por lo que la estimación almacenada de n_1 se propaga a sus sucesores, γ_1 y γ_2 empleando la función $MaxVector$. Se expande γ_1 y se añade al conjunto $COSTS$, eliminando a su vez el coste de γ_3, ya que lo domina. Al estar γ_2 dominado por $COSTS$, se discontinúa de modo definitivo. La estimación almacenada de n_1 se actualiza a ∞, finalizando la búsqueda.

4. Evaluación empírica

Esta sección evalúa el rendimiento de diversas variantes multiobjetivo de $RBFS$. Concretamente se comparan $RIPS$ y $MOMA^*0$. También se ha implementado y comparado una variante *lineal* de $MOMA^*0$, que usa un orden lineal para la expansión. Los experimentos se han realizado sobre árboles binarios infinitos aleatorios [8] con la siguiente configuración: 2 objetivos; soluciones a profundidades 16, 18, 20, 22 y 24 del árbol de búsqueda; rango de costes [1,50] para ambos objetivos; heurístico $\boldsymbol{h}(n) = 0, \forall n$; porcentaje de la cantidad de nodos solución en la profundidad fijada para las soluciones 1 %, 10 %, 25 %, 40 %, 60 %, 80 % y 100 %; 10 problemas aleatorios por cada combinación (profundidad soluciones, cantidad soluciones). Se estableció un límite temporal para los experimentos igual a $5 \times t_{ip}$, donde t_{ip} es el tiempo medio obtenido por $RIPS$ para cada tipo de problemas.

La figura 2 muestra tiempos medios de procesamiento (segundos, escala logarítmica) para cada una de las profundidades fijadas para los nodos solución. No se muestran los experimentos que excedieron el límite temporal fijado. Los algoritmos $MOMA^*0$ y $MOMA^*0$-*lineal* únicamente fueron capaces de solucionar dentro de dicho límite temporal los problemas en los cuales las soluciones se encontraban a un nivel de profundidad 16. Por lo tanto para las profundidades superiores (18, 20, 22 y 24) sólo se muestran los resultados obtenidos por $RIPS$.

El rendimiento de los algoritmos mejora a medida que aumenta la cantidad de soluciones disponibles, ya que esto favorece las podas de la cota global $COSTS$. La principal diferencia entre los algoritmos analizados es la cota de profundización. El rendimiento observado de $MOMA^*0$ y $MOMA^*0$-*lineal* es similar, dado que el coste lineal de un nodo (suma de las componentes de su vector de coste) tampoco permite determinar de modo individual la calidad de dicho nodo para cada uno de los objetivos considerados. La cota empleada por $RIPS$ facilita una profundización más eficiente con respecto a la de $MOMA^*0$. Por último, el hecho de que $RIPS$ incluya temporalmente soluciones subóptimas no influye notoriamente en su rendimiento, presentándose como la mejor opción dentro de las variantes analizadas.



(a) Soluciones prof. 16 (b) Soluciones prof. 18-20-22-24

Figura 2: Comparativa de variantes multiobjetivo de *RBFS*.

5. Conclusiones y trabajo futuro

Los experimentos realizados demuestran que el uso de una cota univectorial calculada como el *punto ideal* de los costes discontinuados, al almacenar de modo sencillo (un vector) información de los mejores valores para cada uno de los objetivos, se convierte en la opción más eficiente a la hora de extender *RBFS* al caso multiobjetivo. Como posibles líneas de trabajo futuro se propone extender la evaluación a la resolución de problemas multiobjetivo reales, así como realizar una comparativa exhaustiva entre algoritmos multiobjetivo basados en *RBFS* y en profundización iterativa. Asimismo sería de gran interés realizar una comparativa entre $RIPS$ y algoritmos multiobjetivo de carácter aproximado.

Referencias

1. Coego, J., Mandow, L., Pérez de la Cruz, J.L.: A comparison of multiobjective depth-first algorithms. Journal of Intelligent Manufacturing (2010)
2. Coego, J., Mandow, L., Pérez de la Cruz, J.L.: Ideal point guided iterative deepening. In: ECAI 2012, LNCS 5883. pp. 246–251 (2012)
3. Coego, J.: Algoritmos de búsqueda con retroceso para problemas multicriterio. Ph.D. thesis, University of Málaga (2015)
4. Dasgupta, P., Chakrabarti, P., DeSarkar, S.: Multiobjective Heuristic Search. Vieweg, Braunschweig/Wiesbaden (1999)
5. Felner, A.: Position paper: The collapse macro in best-first search algorithms and an iterative variant of RBFS. In: Proceedings of 8th SOCS 2015. pp. 28–34 (2015)
6. Harikumar, S., Kumar, S.: Iterative deepening multiobjective A*. Information Processing Letters 58, 11–15 (1996)
7. Korf, R.E.: Linear-space best-first search. Artificial Intelligence 62, 41–78 (1993)
8. Korf, R.E., Chickering, D.M.: Best-first minimax search. Artif. Intell. 84(1-2), 299–337 (1996)
9. Zhang, W.: State-Space Search: Algorithms, Complexity, Extensions, and Applications. Springer (1999)