Metaheurísticas

Seminario 6. Metaheurísticas Multiobjetivo

1. Problemas de Optimización Multiobjetivo
2. Metaheurísticas para Problemas Multiobjetivo
3. Algoritmos Evolutivos Multiobjetivo
4. Elitismo en la Búsqueda Evolutiva Multiobjetivo
5. Conclusiones

Problemas de Optimización Multiobjetivo

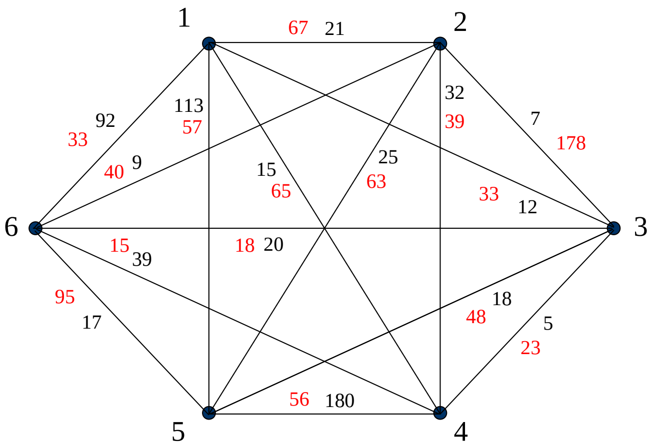
- Muchos problemas reales se caracterizan por la existencia de **múltiples medidas de actuación**, las cuales deberían ser optimizadas, o al menos ser satisfechas simultáneamente

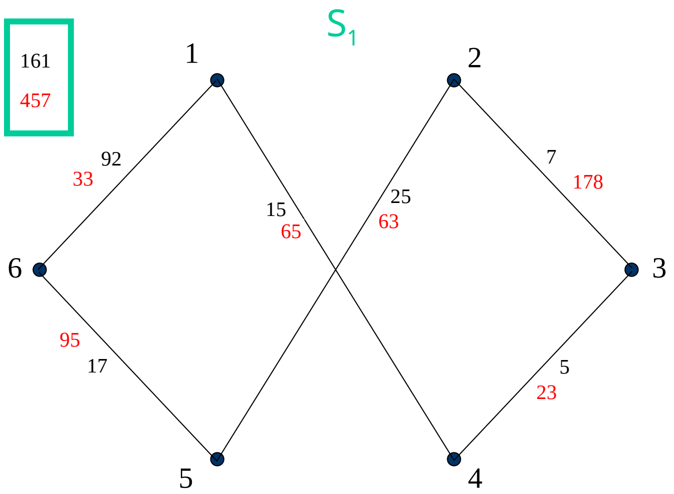
  Ejemplo: Diseño de un sistema de control de aire acondicionado, optimizando el conjunto de parámetros de un sistema de control:

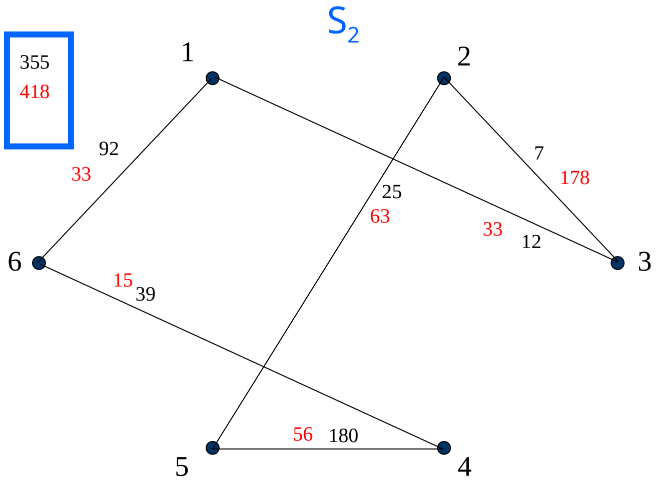
  - Minimizar el consumo de energía
  - Maximizar el confort de los usuarios
  - Maximizar la estabilidad del sistema de control, ...

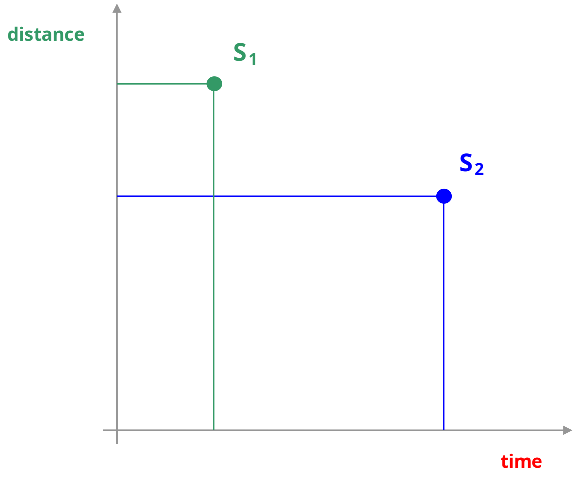
- **Esos objetivos suelen estar en conflicto**, lo que dificulta la resolución del problema

Ejemplo: Viajante de Comercio Multiobjetivo (Minimizar Tiempo y Distancia)









Definición de Problema Multiobjetivo

Un **Problema Multiobjetivo** consiste en: dado un espacio **X** compuesto por vectores n-dimensionales de variables $x=\{x_1,...,x_n\}$ encontrar un vector x^* que minimice (o maximice) un conjunto de K funciones objetivo $z(x) = \{f_1(x),...,f_K(x)\}$ **Y** :

$$\text{Max o Min } \; z(x) = (f_1(x), f_2(x), ..., f_K(x))$$

- **X** es el espacio de decisión (de soluciones)
- **Y** es el espacio objetivo. Normalmente **Y** R^K
- $z(x)=\{f_1,...,f_K\}$ es el conjunto de funciones objetivo
- A veces existen restricciones:
  - Desigualdades: $g_i(x) \leq 0$, $0 \leq i \leq N$.
  - Igualdades: $h_i(x) = 0$, $0 \leq i \leq M$.
  - Otras

Ejemplo: Viajante de Comercio Multiobjetivo (Minimizar Tiempo y Distancia)

- Ejemplo: TSP con distancias y tiempos en cada arco
  - **X = C^{n}**,
    - **C** es el conjunto de ciudades
    - **n** es el número de ciudades
  - **Y** $\square$ R^2
  - Funciones objetivo:
    - $f_1(x)$ = tiempo empleado en recorrer el tour x
    - $f_2(x)$ = longitud del tour x
  - Restricciones:
    - $x_i \square x_j$; $0 \leq i,j \leq n$; $i \square j$

Concepto de Dominancia

Un **Problema Multiobjetivo** consiste en:

$$\text{Max o Min} \quad z = f(x) = (f_1(x), f_2(x), \ldots, f_K(x))$$

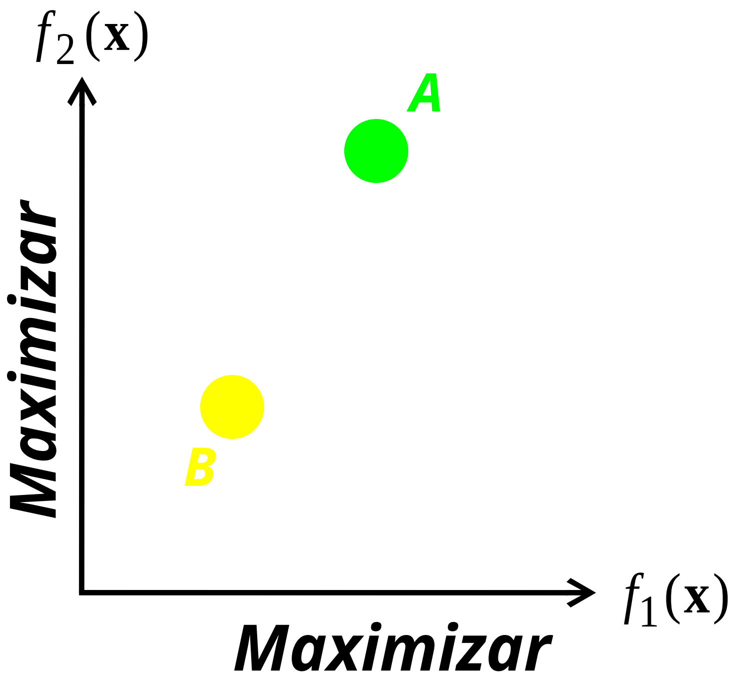
Soluciones pareto-optimales o no-dominadas: Se dice que un vector ***a*** domina a otro ***b*** (se nota como $\boldsymbol{a \succ b}$) si, y sólo si (suponiendo maximización):

$$\forall i \in \{1,2,\ldots,K\} : f_i(a) \geq f_i(b) \wedge \exists\, j \in \{1,2,\ldots,K\} : f_j(a) > f_j(b)$$

Es decir, una solución domina a otra si es mejor o igual en todos los objetivos y mejor en al menos uno de ellos

Concepto de Dominancia

Maximizar $$\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ f_2(\mathbf{x}))$$



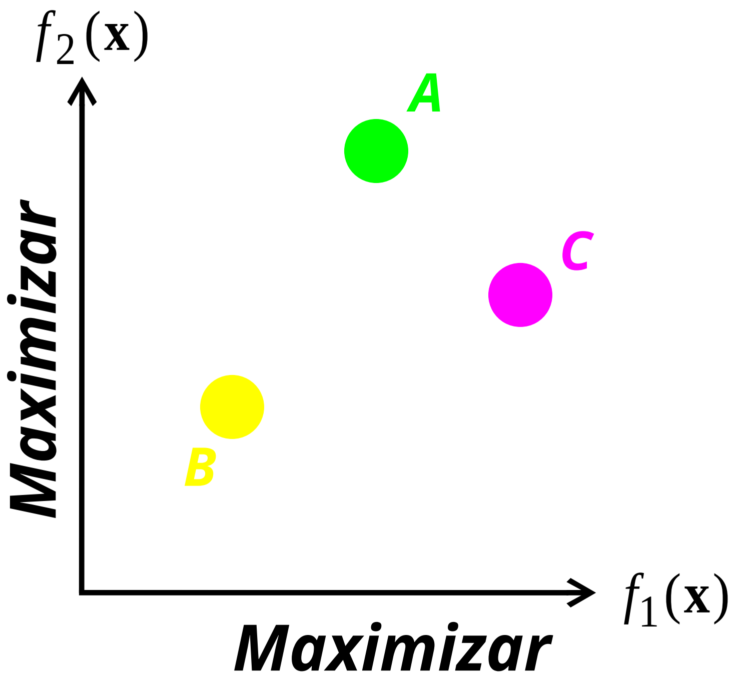
A domina a B

B es dominada por A

(A es mejor que B)

Concepto de Dominancia

Maximizar $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ f_2(\mathbf{x}))$



A y C son no dominadas entre sí (ninguna domina a la otra)

Las dos dominan a B

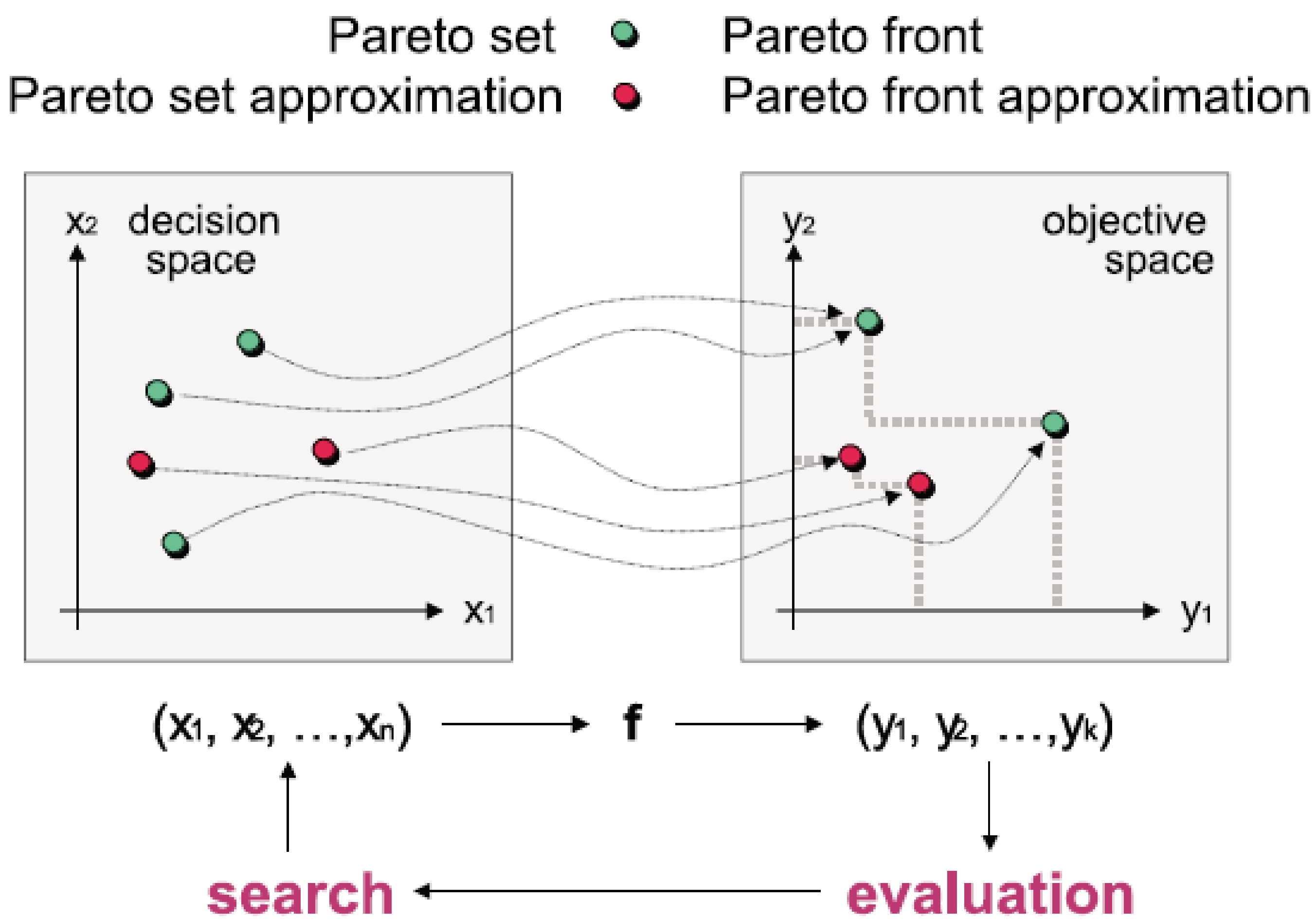
Concepto de Dominancia

Una solución es **Pareto-optimal** si no es dominada por ninguna otra solución del espacio

El conjunto de todas las soluciones no dominadas $X^* \subseteq \mathbf{X}$ es el **conjunto Pareto-optimal** y compone la **solución óptima del problema multiobjetivo**

Los vectores de valores de las funciones objetivo de los elementos del conjunto Pareto-optimal $z(X^*) \subseteq \mathbf{Y}$ forman la **frontera** (o el **frente) de Pareto**

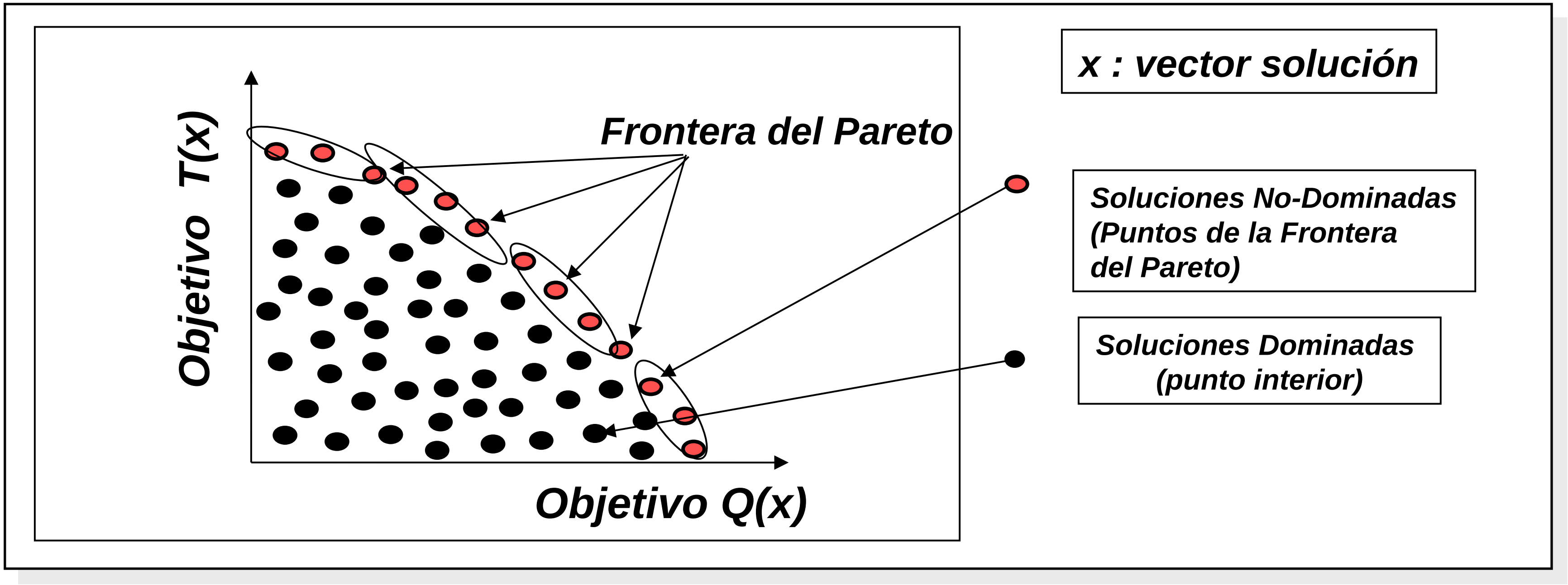
Ejemplo: Conjuntos de Pareto en $\mathbb{R}^2$



Objetivo de la Optimización Multiobjetivo

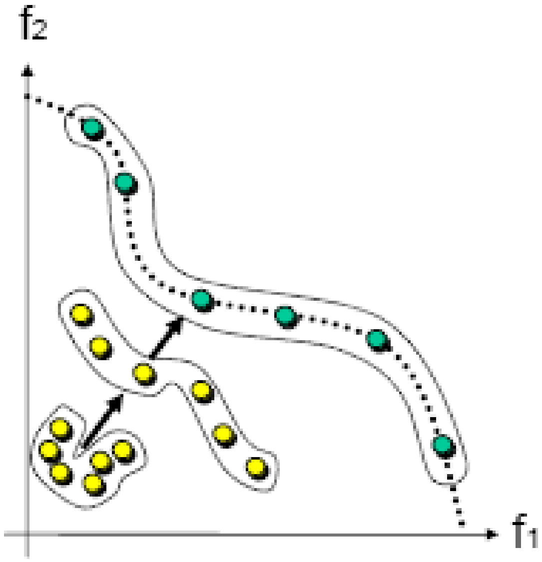
- No suele existir una única solución optimal, existe un conjunto (a veces infinito) de soluciones No-Dominadas que forma la Frontera del Pareto
  - Ejemplo:

Identificar la frontera del Pareto para [Max Q(x), Max T(x)]



Objetivo de la Optimización Multiobjetivo

- El objetivo es encontrar una aproximación del frente de Pareto de la mayor calidad posible
  - Debe estar tan cerca del frente de Pareto óptimo como sea posible
  - Las soluciones deben estar uniformemente distribuidas sobre el frente
  - La aproximación debe capturar todo el frente del Pareto, incluyendo los extremos

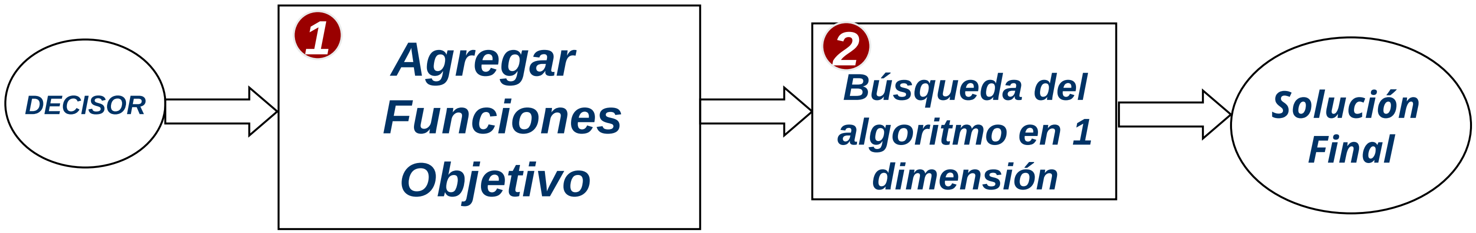


Resolución de Problemas Multiobjetivo

¿Qué necesitamos para resolver este problema?:

- *Un método de búsqueda basado en los múltiples objetivos*
- *Una política de equilibrio entre los objetivos*
- *Un orden para este proceso de optimización*

Vamos a considerar 2 posibilidades: a) Agregación + búsqueda



Primero se agregan los objetivos en una única función o se considera un orden entre ellos dando lugar a una función adecuada para el algoritmo mono-objetivo que se aplica a continuación

Resolución de Problemas Multiobjetivo

- Funciones de agregación: Se agregan todos los objetivos en una única función objetivo:
  - $f(x) = F(f_1(x),\dots,f_K(x))$

- Orden lexicográfico: Se considera un orden jerárquico para los objetivos:
  - Primero min $f_1(x)$, luego min $f_2(x)$,….
  - Para comparar dos soluciones se mira primero $f_1(x)$. En caso de empate, se mira $f_2(x)$ y así sucesivamente
  - Ejemplo: TSP con tiempo y distancia. Encontrar el tour con la menor distancia de entre los tours de menor tiempo

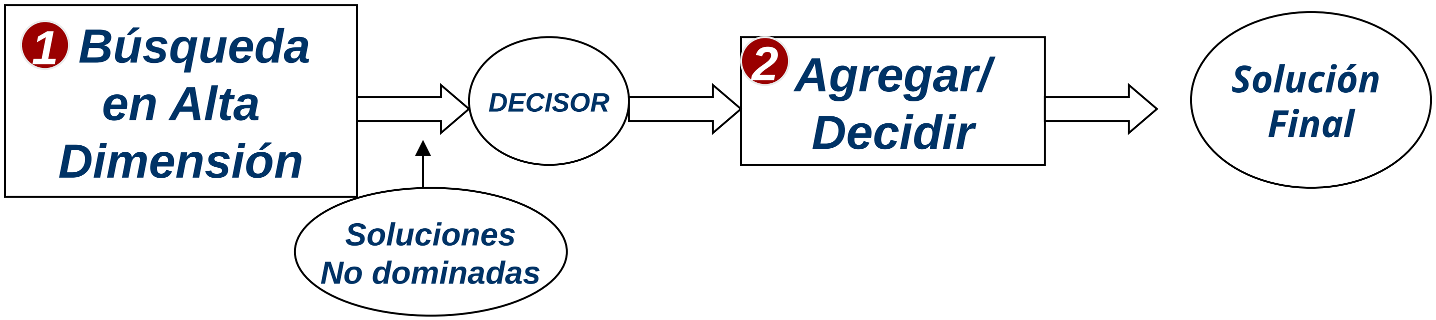
En cualquier caso, el algoritmo mono-objetivo se tiene que ejecutar muchas veces para obtener una aproximación del frente de Pareto

Resolución de Problemas Multiobjetivo

¿Qué necesitamos para resolver este problema?:

- *Un método de búsqueda basado en los múltiples objetivos,*
- *Una política de equilibrio entre los objetivos,*
- *Un orden para este proceso de optimización*

Vamos a considerar 2 posibilidades: b) Búsqueda + agregar/decidir



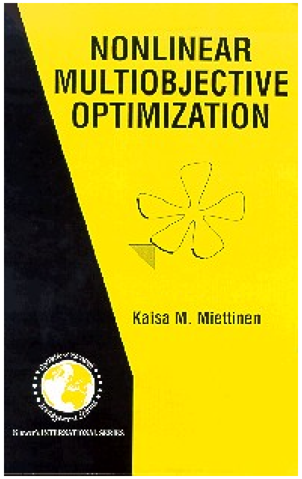
Nota: Se puede considerar una tercera posibilidad híbrida, combinando búsqueda en alta dimensión con búsquedas en dimensiones menores vía agregación parcial de objetivos, como modelos interactivos.

Metaheurísticas para Problemas Multiobjetivo

- En la actualidad, hay unas 30 técnicas clásicas de programación matemática para resolver problemas de optimización multiobjetivo (MO)

- Sin embargo, **estas técnicas suelen generar los elementos del conjunto de Pareto de uno en uno**, requiriendo múltiples ejecuciones para obtener una aproximación

- Además, muchas de ellas son muy sensibles a la forma del frente de Pareto (p.e., no funcionan cuando el frente es cóncavo o está desconectado)



Metaheurísticas para Problemas Multiobjetivo

- Por estas razones, la mayoría de los enfoques MO existentes están basados en metaheurísticas (MHs), en particular en algoritmos evolutivos (AEs) (70%)

- La mayor parte de ellos (un 90%) aplican el **segundo enfoque**, tratando de obtener una buena aproximación del frente de Pareto

- Los AEs son muy buenos optimizadores MO debido a que:
  - Son algoritmos poblaciones que permiten obtener múltiples soluciones en una única ejecución
  - Se adaptan a buscar en distintas zonas del espacio simultáneamente
  - No son sensibles a la forma del frente de Pareto

Algoritmos Evolutivos Multiobjetivo

- Se evoluciona una población de soluciones al problema

- Se aplican mecanismos que mantengan diversidad en la población para conseguir un conjunto de soluciones no dominadas lo más grande posible

- Dos tipos de modelos de acuerdo a las tipologías a) y b):

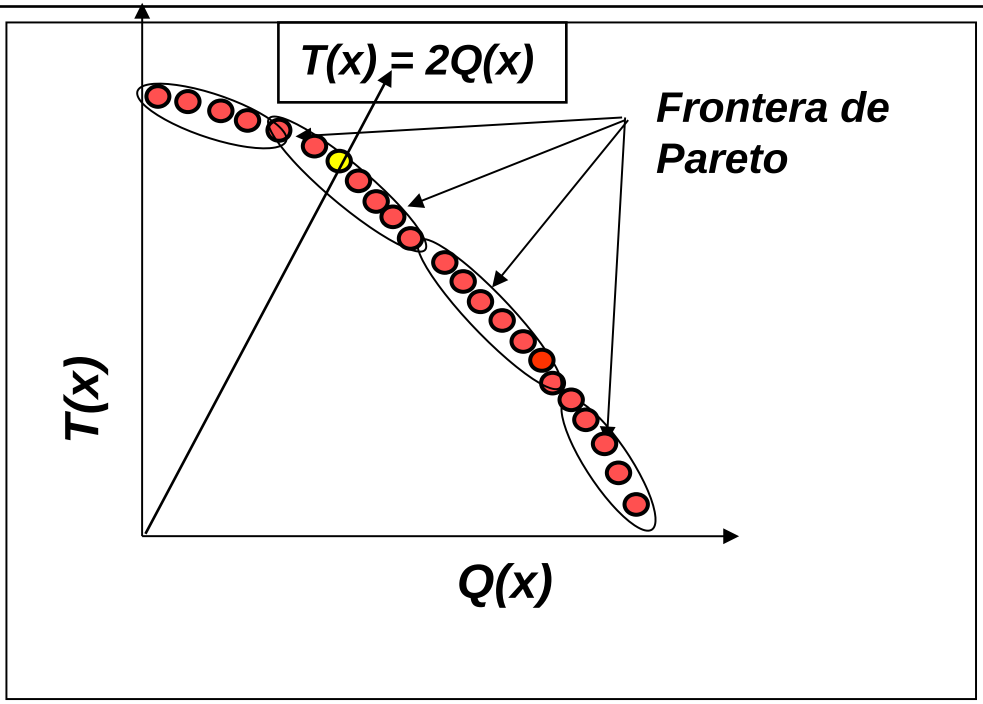
  - **Modelos evolutivos utilizando pesos para la agregación de los objetivos**

  - **Modelos evolutivos que generan poblaciones de soluciones no dominadas**

Modelos Evolutivos utilizando Pesos

- La agregación de los objetivos conduce a la obtención de un único punto de equilibrio en la frontera
- Ejemplo: *[Max Q(x), Max T(x)]*

  Dar a T(x) dos veces la importancia de Q(x), ej: T(x) = 2*Q(x)



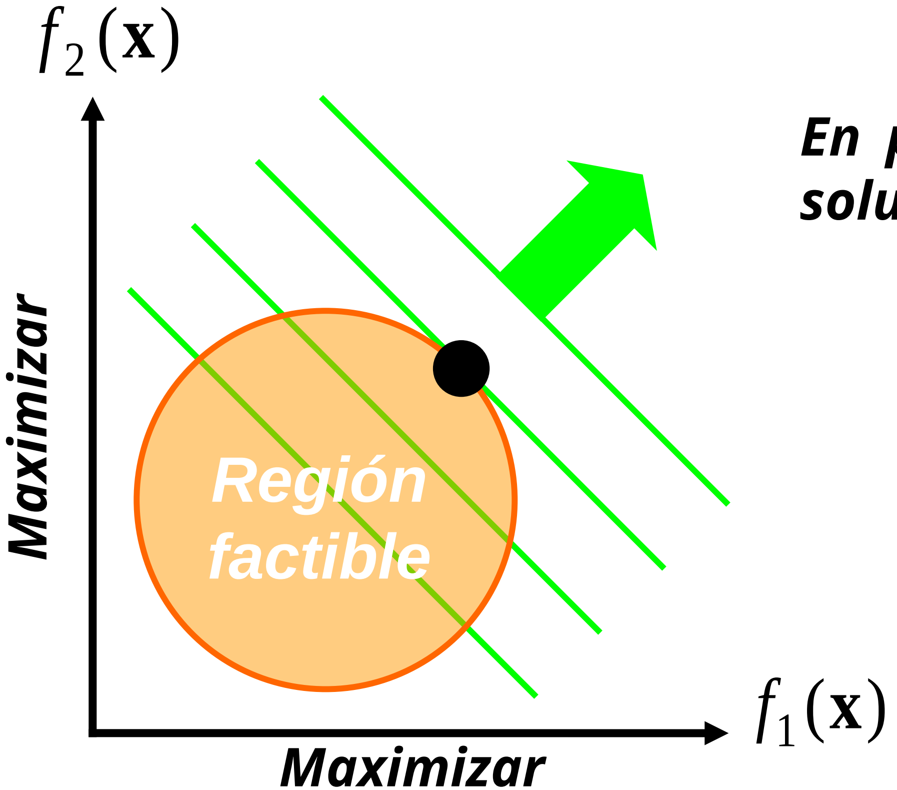
La línea T(x) = 2*Q(y) corresponde al vector de pesos W: [1 , 2], cuando se utiliza una función que combina ambos objetivos

F = W * [Q(x), T(X)]
F = [1, 2] * [Q(x), T(X)]
F = Q(x) + 2*T(x)

Modelos Evolutivos utilizando Pesos (2)

- Presentan los problemas habituales de un optimizador MO basado en agregación de los objetivos usando pesos:

$$\textit{Maximizar} \quad f(x) = w_1 f_1(x) + w_2 f_2(x)$$

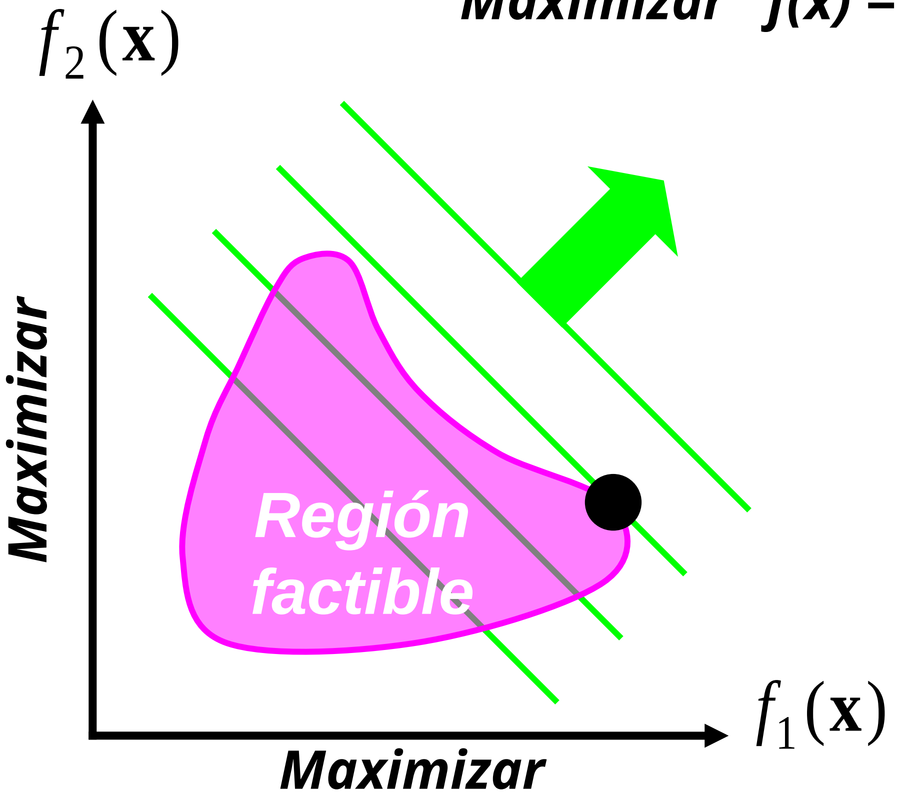


En principio, se obtiene una única solución en cada ejecución

Modelos Evolutivos utilizando Pesos (3)

- El enfoque es muy sensible a la especificación de los pesos
- No puede encontrar soluciones en regiones cóncavas del frente de Pareto

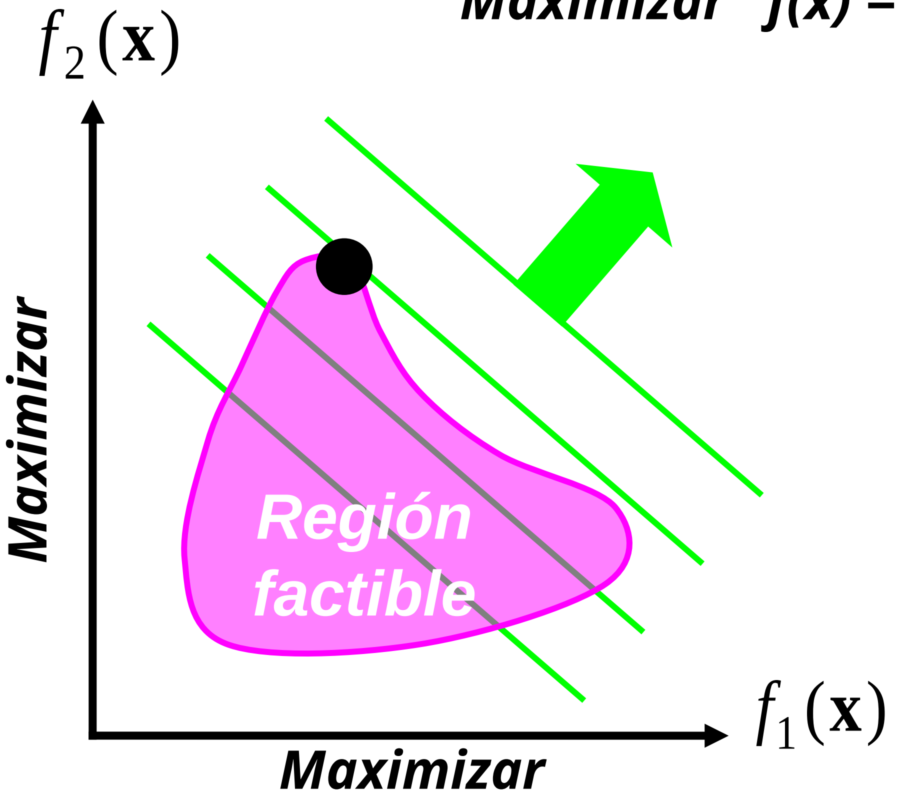
Maximizar $f(x) = w_1 f_1(x) + w_2 f_2(x)$



Modelos Evolutivos utilizando Pesos (4)

- El enfoque es muy sensible a la especificación de los pesos
- No puede encontrar soluciones en regiones cóncavas del frente de Pareto

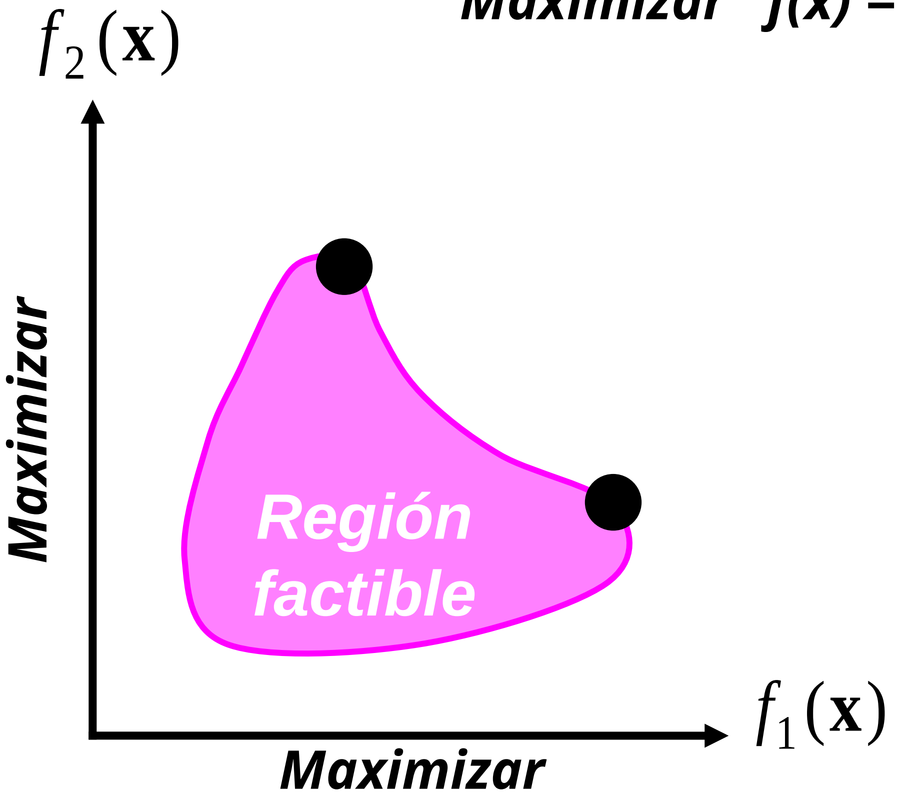
Maximizar $f(x) = w_1 f_1(x) + w_2 f_2(x)$



Modelos Evolutivos utilizando Pesos (5)

- El enfoque es muy sensible a la especificación de los pesos
- No puede encontrar soluciones en regiones cóncavas del frente de Pareto

Maximizar $\boldsymbol{f(x) = w_1 f_1(x) + w_2 f_2(x)}$



Modelos Evolutivos utilizando Pesos (6)

- *VOW-GA: Variable Objective Weighting GA (Hajela & Lin 1992)*
- *RW-GA: Random Weights GA*
  (Ishibuchi & Murata, 1998)

Se basan en trabajar con varios vectores de pesos, ya sean aprendidos por el algoritmo genético (codificados en el propio cromosoma) o aleatorios para cada evaluación

Gracias a ello, pueden obtener varias soluciones del frente del Pareto en una sola ejecución

Modelos Evolutivos que Generan Poblaciones de Soluciones No Dominadas

Primera generación de Algoritmos Evolutivos Multiobjetivo basados en soluciones no-dominadas

- ***MOGA: Multi-objective Optimization GA***

  C.M. Fonseca, P.J. Fleming, Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. S. Forrest (Ed.), Proc. 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, 1993, 416-423.

- ***NPGA: Niched Pareto GA***

  J. Horn, N. Nafpliotis. Multiobjective Optimization Using the Niched Pareto Genetic Algorithms. IlliGAL Report 93005, University of Illinois, Urbana, Champaign, July 1993.

- ***NSGA: Non-dominated Sorting GA***

  N. Srinivas, K. Deb, Multiobjetive Optimization Using Nondominated Sorting in Genetic Algorithms. Evolutionary Computation 2 (1995) 221-248.

Modelos Evolutivos que Generan Poblaciones de Soluciones No Dominadas (2)

- ***MOGA: Multi-objective Optimization GA** (Fonseca & Fleming 1993)*

  C.M. Fonseca, P.J. Fleming, Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. S. Forrest (Ed.), Proc. 5th Int. Conf. on Genetic Algorithms, Morgan Kaufmann, 1993, 416-423.

  A cada individuo de la población se le asignará un rango de acuerdo al cual será ordenado para la selección

  El rango se asigna según un criterio de no dominancia

  Si x_i es no dominado entonces rango(x_i) = 1
  En otro caso rango(x_i) = 1 + (no. de individuos que lo dominan)

Modelos Evolutivos que Generan Poblaciones de Soluciones No Dominadas (3)

- ***MOGA: Multi-objective Optimization GA***
  (Fonseca & Fleming 1993)

  Una vez calculado el rango de los individuos de la población se siguen los siguientes pasos:

  1. La población se ordena de menor a mayor de acuerdo al rango que se le ha asignado a cada individuo

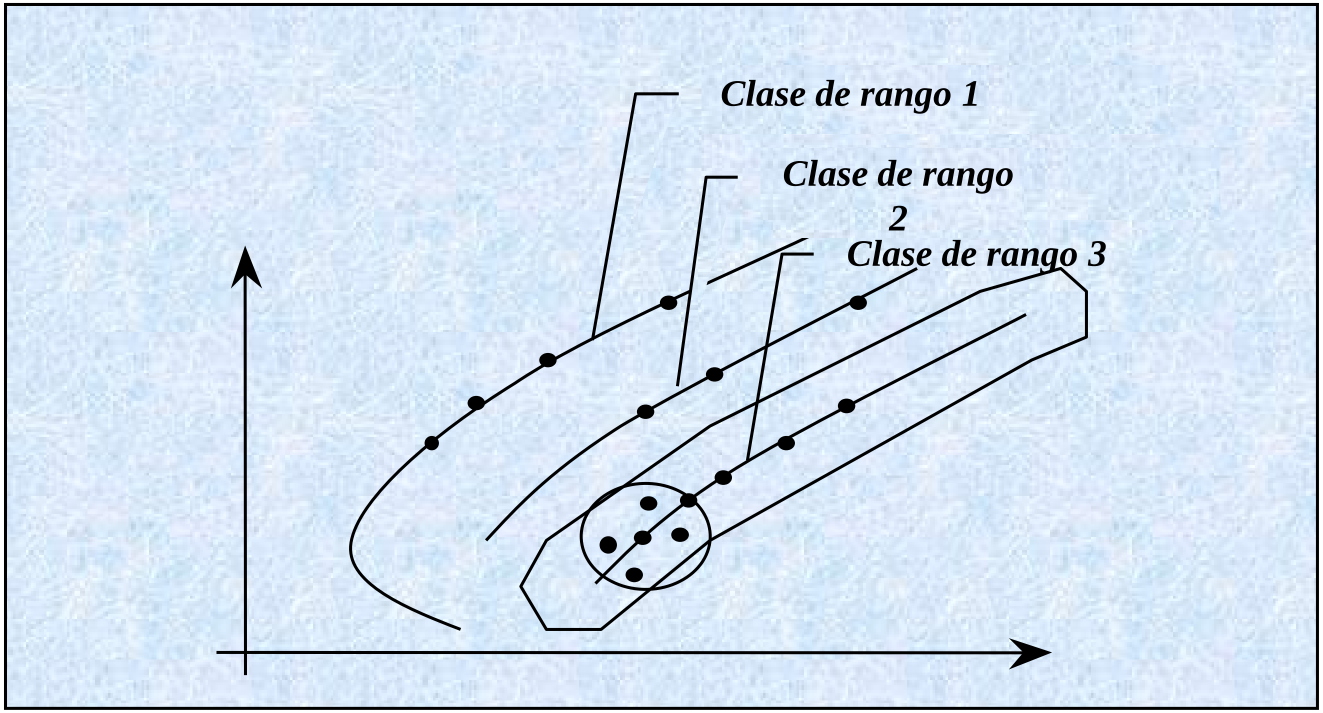
  2. Se asigna el valor de adaptación para cada individuo por interpolación desde el mejor (rango 1) hasta el peor

  3. Se promedia la adaptación de los individuos con el mismo rango, para que tengan el mismo valor de adaptación

  Nuevas versiones utilizan técnicas de proporción de nichos (sharing) sobre los objetivos

Modelos Evolutivos que Generan Poblaciones de Soluciones No Dominadas (3)

- ***MOGA: Multi-objective Optimization GA** (Fonseca & Fleming 1993)*



Elitismo en la Búsqueda Evolutiva Multiobjetivo. Conjunto Elite

SPEA y SPEA2

Elitismo Dentro de la Población: NSGA II

Elitismo en la Búsqueda Evolutiva Multiobjetivo. Conjunto Elite

Los AEs MO de segunda generación incorporan el concepto de elitismo

El modelo más extendido usa una población externa, donde se almacenan soluciones no-dominadas encontradas a lo largo de la búsqueda

Esto permite al algoritmo cubrir de un modo más adecuado el Frente del Pareto

Este conjunto de soluciones no dominadas se suele llamar "conjunto elite", P^e, con tamaño N^e. Su uso lleva asociadas dos cuestiones:

Población → Conjunto elite

¿Qué soluciones de P se mantienen en P^e?

Conjunto elite → Población

¿Cómo y cuando los elementos de P^e se reinsertan en P?

Ejemplos:** **Modelos elitistas SPEA y SPEA2

Elitismo en la Búsqueda Evolutiva Multiobjetivo. Conjunto Elite (2)

Modelo Genérico de Algoritmo Evolutivo Multiobjetivo Elitista

$t := 0$

$(A^0, B^0, p_e^0) := inicializar$

MIENTRAS $terminar(A^t, B^t, t) = falso$ **HACER**

$\quad t := t + 1$

$\quad A^t := truncar(actualizar(A^{t-1}, B^{t-1}))$

$\quad p_e^t := adaptar(A^t, B^{t-1}, p_e^{t-1})$

$\quad B^t := operadores(selección(evaluación(A^t, B^{t-1}, p_e^t)))$

Fin MIENTRAS

A^t : **población élite** $\qquad$ B^t : **población**

p_e^t := **intensidad del elitismo en la generación t.**

SPEA: Modelo Evolutivo con Conjunto Elite

- ***STRENGTH PARETO EVOLUTIONARY ALGORITHMS (SPEA) (Zitzler, Thiele, 1998)***

Zitzler, E., Thiele, L. (1998a) An evolutionary algorithm for multiobjective optimization: The strength Pareto Approach. Technical Report 43, Zürich, Switzerland: Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH)

E. Zitzler, L. Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation 3:4 (1999) 257-217

Zitzler, E., Deb, K., Thiele, L. (2000) Comparison of multiobjective evolutionary algorithms: Empirical results. Evolutionary Computation Journal 8(2), 125-148

Este algoritmo introduce el concepto de CONJUNTO ELITISTA EXTERNO DE SOLUCIONES NO DOMINADAS, actualmente muy importante en el desarrollo de AEs MO

SPEA (2)

Algoritmo SPEA

Paso 1. Generar la población inicial P y el conjunto P^e vacío

Paso 2. Copiar las soluciones no dominadas de P en P^e

Paso 3. Quitar en P^e aquellas soluciones dominadas por otras

Paso 4. Si $|P^e| > N^e$, entonces reducir el conjunto a tamaño N^e mediante técnicas de clustering

Paso 5. Calcula el fitness de los individuos de $P' = P + P^e$

Paso 6. Seleccionar N individuos a partir de P' (mediante torneo binario)

Paso 7. Aplicar cruce y mutación

Paso 8. Volver al Paso 2 si no se ha alcanzado el máximo número de iteraciones

SPEA2

Eckart Zitzler, Marco Laumanns, Lothar Thiele: SPEA2: Improving the Strength Pareto Evolutionary Algorithm.
Zürich, TIK Report Nr. 103, Computer Engineering and Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zurich, May, 2001.

Eckart Zitzler

https://sop.tik.ee.ethz.ch/people/zitzler/

https://scholar.google.ch/citations?user=GW8tPekAAAAJ&hl=de

Source code

https://sop.tik.ee.ethz.ch/download/supplementary/testProblemSuite/?page=testProblem.php#source

https://sop.tik.ee.ethz.ch/download/supplementary/testProblemSuite/

PISA

A Platform and Programming Language Independent Interface for Search Algorithms

https://sop.tik.ee.ethz.ch/pisa/

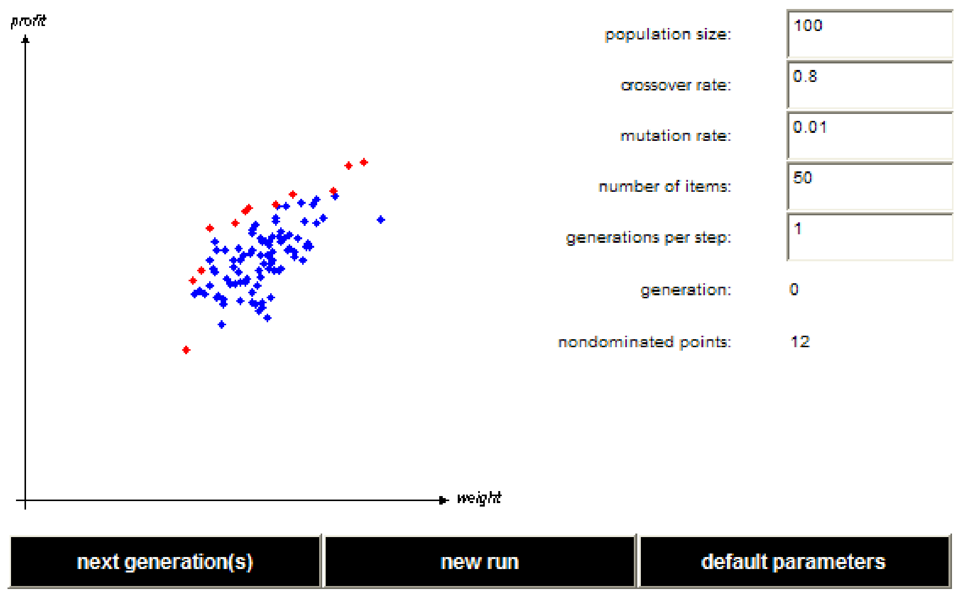
SPEA2 (2)

https://sop.tik.ee.ethz.ch/education/misc/moeaApplet/

A Multiobjective Evolutionary Algorithm for the 0/1 Knapsack Problem

Author: Eckart Zitzler

This page contains a Java applet that implements a simple evolutionary algorithm for the multiobjective 0/1 knapsack problem. It demonstrates how a set of optimal trade-offs between overall profit and weight is approximated by an iterative process: a set of solution candidates undergoes selection, recombination, and mutation to generate a new (and hopefully better) set of solution candidates.



Clasificación de los AEs MO con Conjunto Élite

Características que permiten clasificar a los MOEA Elitistas:

- *Utilizan una población secundaria "elite": A^t*
- *Estrategia de elitismo: ¿Cómo se actualiza la población elitista, A^t?*
- *Estrategia de evaluación: ¿Cómo afectan los individuos del conjunto elite a la asignación de fitness en la población y viceversa?*
- *Estrategia de reinserción: ¿Cómo toman parte los individuos elite en el proceso de reproducción de descendientes?*
- *Los diferentes MOEA elitistas difieren, entre otras cosas, en la aplicación de estas tres estrategias*

NSGA-II: Modelo Evolutivo con Elitismo dentro de la Población Genética

***NSGA-II:* K. Deb, A. Pratap, S. Agarwal and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6:2 (2002) 182-197.**

Muy extendido y considerado por muchos el mejor modelo

- *Nondominated Sorting Genetic Algorithm II (NSGA-II) fue propuesto por Kalyan Deb y sus estudiantes en 2000*
- *Es una versión mejorada del NSGA que utiliza un operador de crowding que no requiere parámetros en vez de usar nichos*
- *NSGA-II utiliza un esquema de selección más en el cual la población de padres se compara con la población de hijos*
- *Además de contar con el uso de elitismo, NSGA-II es mucho más eficiente (computacionalmente) que NSGA y es un algoritmo altamente competitivo en convergencia al Pareto*

NSGA-II (2)

- *En cada generación, se crea un conjunto mediante la unión de la actual población y la población de hijos generada mediante selección, cruce y mutación*
- *De este conjunto se extraen los diferentes frentes (agrupados según el número de soluciones que los dominan). El frente F_1 coincide con la aproximación actual del frente de Pareto óptimo*
- *La nueva población se crea incluyendo los frentes (de mejor a peor) hasta alcanzar el tamaño máximo. Si no caben todos los individuos del último frente, se trunca atendiendo al orden basado en el crowding (medida de concentración)*

$R_t = P_t \cup Q_t$	combine parent and offspring population
$\mathcal{F} = \texttt{fast-non-dominated-sort}(R_t)$	$\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \ldots)$, all nondominated fronts of R_t
$P_{t+1} = \emptyset$ and $i = 1$	
until $\lvert P_{t+1}\rvert + \lvert\mathcal{F}_i\rvert \leq N$	until the parent population is filled
$\quad\texttt{crowding-distance-assignment}(\mathcal{F}_i)$	calculate crowding-distance in $\mathcal{F}_i$
$\quad P_{t+1} = P_{t+1} \cup \mathcal{F}_i$	include ith nondominated front in the parent pop
$\quad i = i + 1$	check the next front for inclusion
$\text{Sort}(\mathcal{F}_i, \prec_n)$	sort in descending order using $\prec_n$
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i[1 : (N - \lvert P_{t+1}\rvert)]$	choose the first $(N - \lvert P_{t+1}\rvert)$ elements of $\mathcal{F}_i$
$Q_{t+1} = \texttt{make-new-pop}(P_{t+1})$	use selection, crossover and mutation to create a new population Q_{t+1}
$t = t + 1$	increment the generation counter

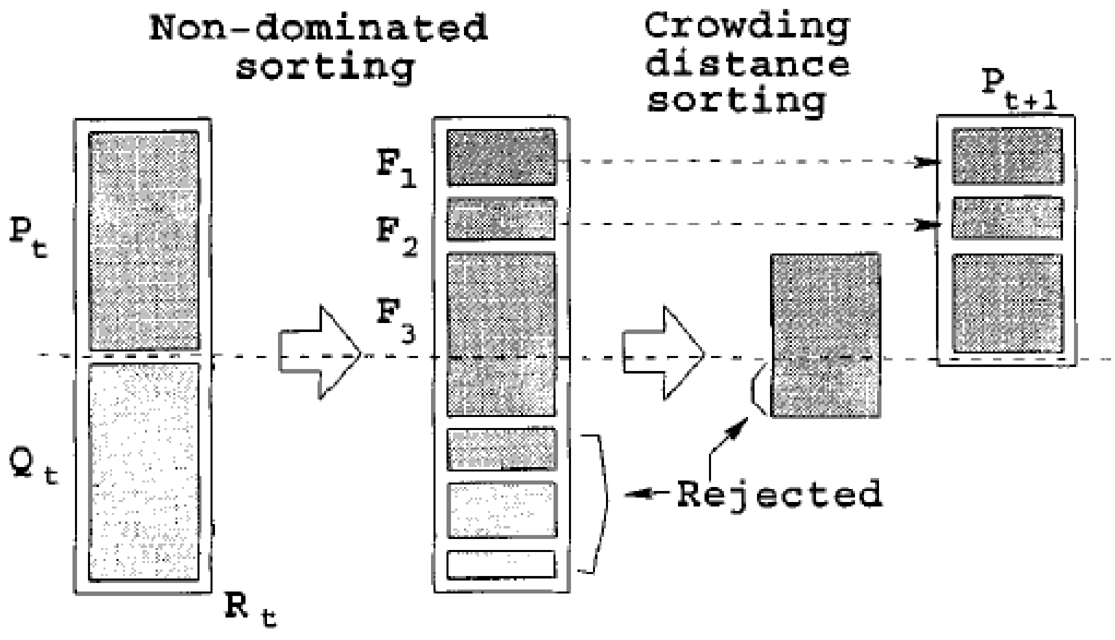
NSGA-II (3)

```
fast-non-dominated-sort(P)
for each p ∈ P
  S_p = ∅
  n_p = 0
  for each q ∈ P
    if (p ≺ q) then                    If p dominates q
      S_p = S_p ∪ {q}                  Add q to the set of solutions dominated by p
    else if (q ≺ p) then
      n_p = n_p + 1                    Increment the domination counter of p
  if n_p = 0 then                      p belongs to the first front
    p_rank = 1
    F_1 = F_1 ∪ {p}
i = 1                                  Initialize the front counter
while F_i ≠ ∅
  Q = ∅                                Used to store the members of the next front
  for each p ∈ F_i
    for each q ∈ S_p
      n_q = n_q − 1
      if n_q = 0 then                  q belongs to the next front
        q_rank = i + 1
        Q = Q ∪ {q}
  i = i + 1
  F_i = Q
```

NSGA-II (4)

- *La medida de crowding se utiliza para seleccionar las soluciones más dispersas entre los individuos del último frente utilizado en la nueva población (F_3 en este ejemplo)*

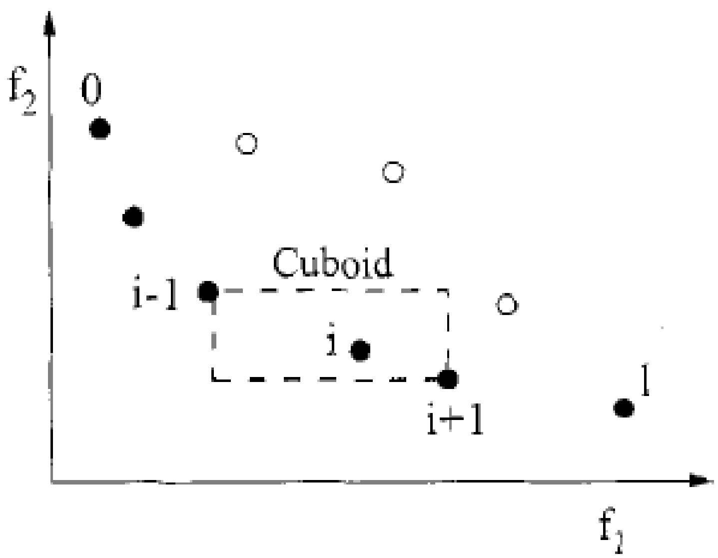
- *Cuanto mayor sea la distancia de crowding de una solución al resto de su frente mejor, ya que hay menos concentración en esa zona*



NSGA-II (5)

crowding-distance-assignment($\mathcal{I}$)

$l = \lvert\mathcal{I}\rvert$	number of solutions in $\mathcal{I}$
for each i, set $\mathcal{I}[i]_{\text{distance}} = 0$	initialize distance
for each objective m	
$\quad \mathcal{I} = \text{sort}(\mathcal{I}, m)$	sort using each objective value
$\quad \mathcal{I}[1]_{\text{distance}} = \mathcal{I}[l]_{\text{distance}} = \infty$	so that boundary points are always selected
$\quad$ for $i = 2$ to $(l - 1)$	for all other points
$\quad\quad \mathcal{I}[i]_{\text{distance}} = \mathcal{I}[i]_{\text{distance}} + (\mathcal{I}[i+1].m - \mathcal{I}[i-1].m)/(f_m^{\max} - f_m^{\min})$	



NSGA-II (6)

Problema de la Mochila bi-objetivo con 500 objetos (Maximización)

200 soluciones aleatorias y el frente de Pareto óptimo

NSGA-II (7)

Problemas:

- ***Parece tener un comportamiento más pobre cuando se utiliza con representación binaria***
- ***Tiende a tener problemas exploratorios conforme se incrementa el número de funciones objetivo (lo tienen todos los algoritmos actuales) ***

NUEVOS MOEAs: MANY-OBJECTIVIZATION

NSGA-II (8)

Kalyanmoy Deb
http://www.iitk.ac.in/kangal/
https://www.egr.msu.edu/~kdeb/

The IEEE TEVC paper describing NSGA-II for multi-objective optimization is judged as the FAST-BREAKING PAPER IN ENGINEERING by Web of Science (ESI) in February 2004

Softwares Developed at KanGAL
http://www.iitk.ac.in/kangal/codes.shtml

- *Multi-objective NSGA-II code in C*
  - *Original Implementation (for Windows and Linux): NSGA-II in C (Real + Binary + Constraint Handling)*
  - *New (10 April 2005) (for Linux only): NSGA-II in C (Real + Binary + Constraint Handling)*
  - *Revision 1.1 (10 May 2005) (for Linux only): NSGA-II in C (Real + Binary + Constraint Handling)*
  - *Revision 1.1 (10 June 2005) (for Linux only): NSGA-II in C with gnuplot (Real + Binary + Constraint Handling)*

NSGA-II (9)

Prof. Deb receives Shanti Swarup Bhatnagar Prize

from Honorable Prime Minister of India

on 28 September 2005 in New Delhi

Conclusiones

- *Los MOEAs se han convertido en una de las áreas de investigación más activas en el ámbito de la Computación Evolutiva*

- *Su aplicabilidad en los problemas multiobjetivo es clara e inmediata, y se han convertido en una herramienta muy importante para abordar dichos problemas*

- *Es un área consolidada y a la vez muy abierta desde la doble perspectiva de la investigación en el desarrollo de nuevos MOEAs (incorporación de preferencias, funciones dinámicas, espacios con restricciones, escalabilidad en cuanto al numero de objetivos, trade-off eficiencia-eficacia en problemas complejos, paralelismo, …), como de la aplicación*

- ***La comparación de las soluciones obtenidas por un MOEA (de las aproximaciones del frente de Pareto) es un problema complejo***

Conclusiones (2)

¿Qué aproximación del frente de Pareto es mejor?

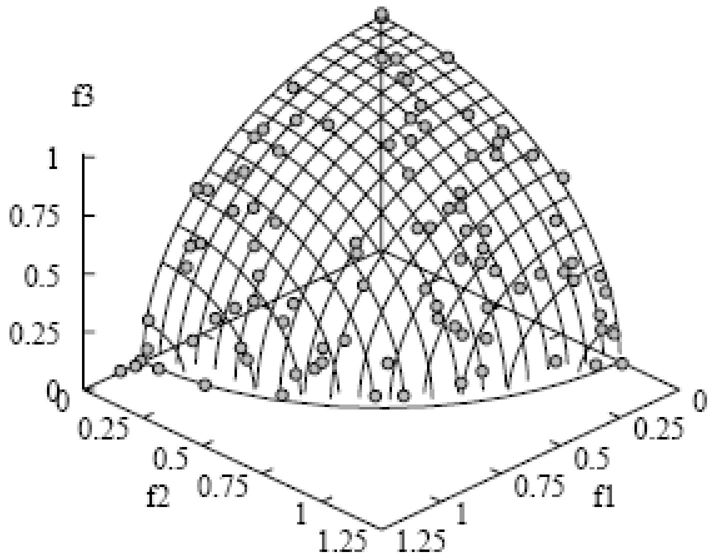


Figure 9: Non-dominated points obtained using NSGA-II.

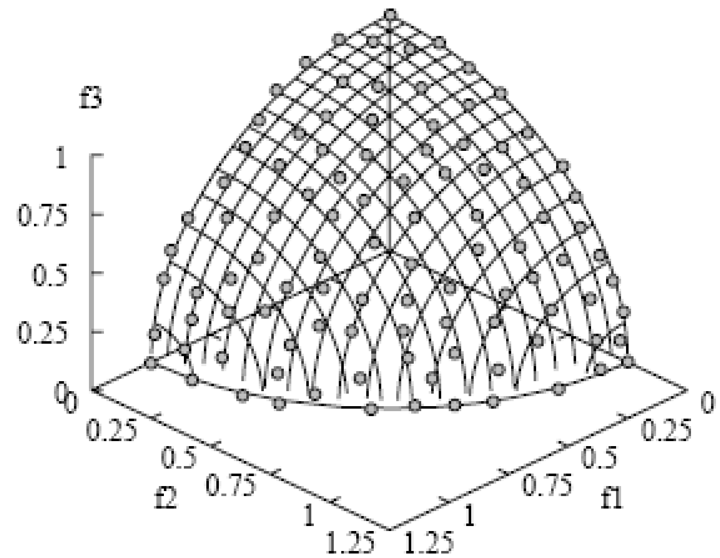
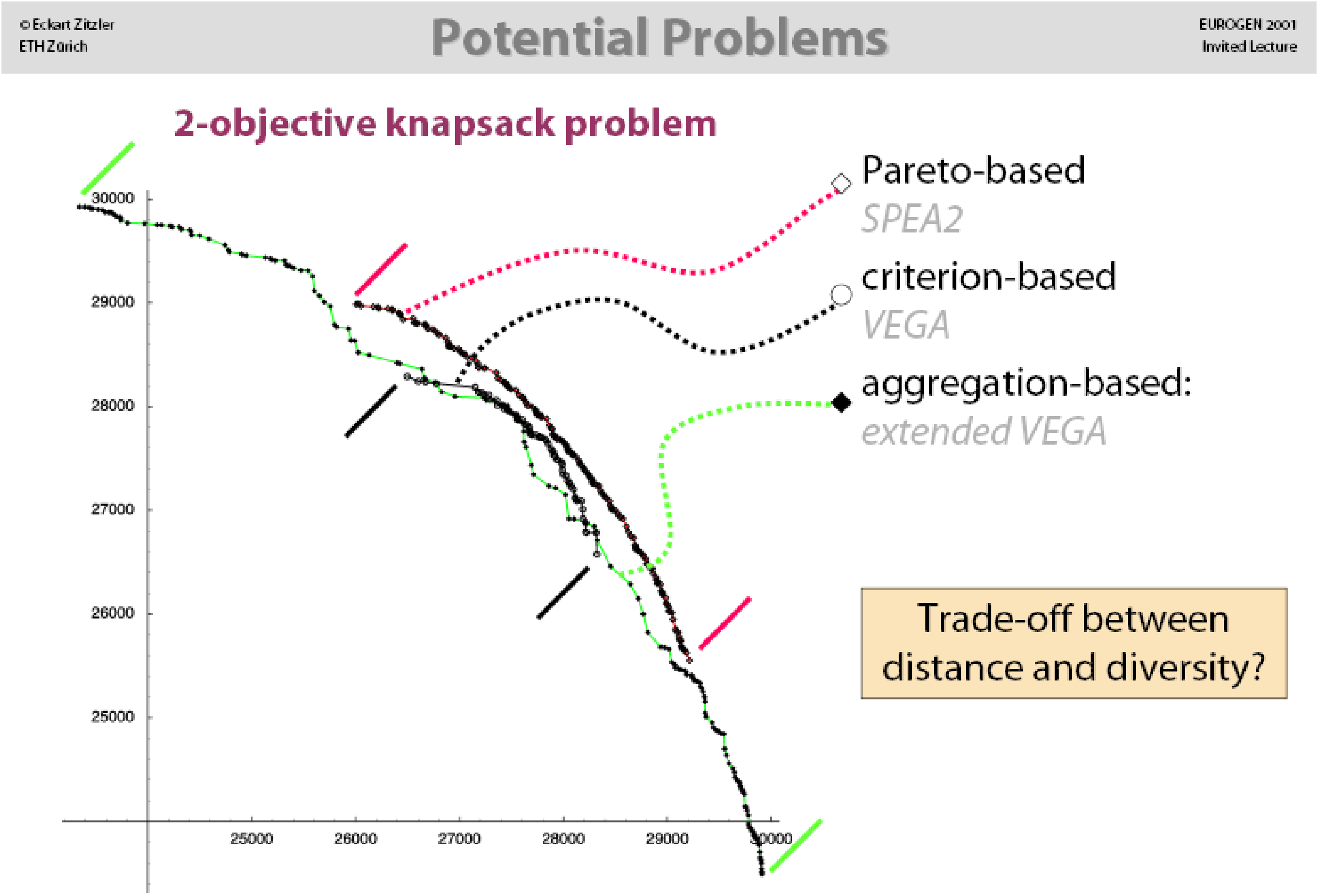


Figure 10: Non-dominated points obtained using SPEA2.

Conclusiones (3)

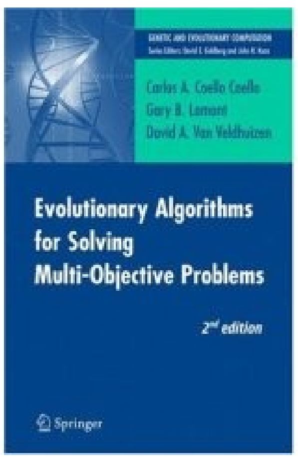


Más sobre AEs MO

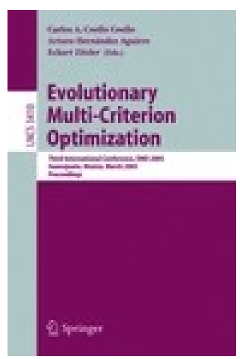
PARA SABER MAS SOBRE OPTIMIZACIÓN EVOLUTIVA MULTIOBJETIVO

Visitar el repositorio de EMOO localizado en:

http://delta.cs.cinvestav.mx/~ccoello/EMOO



C.A. Coello, D.A. Van Veldhuizen, G.B. Lamont, Evolutionary Algorithms for Solving Multi-Objective Problems. Kluwer Academic Pub., 2007 (second-edition) .



Evolutionary Multi-Criterion Optimization
Third Int. Conf, EMO 2005, Guanajuato, Mexico, March 9-11, 2005, Proceedings
Series: Lecture Notes in Computer Science, Vol. 3410
Coello Carlos A.; Hernández, Arturo; Zitzler, Eckart (Eds.) 2005, XVI, 912 p.,

C.A. Coello
https://scholar.google.com/citations?user=oJMnjNYAAAAJ&hl=es

Bibliografía

Lecturas Básicas:

C.A. Coello. Evolutionary Multiobjective Optimization: Current and Future Challenges. In J. Benitez, O. Cordon, F. Hoffmann, and R. Roy (Eds.), Advances in Soft Computing---Engineering, Design and Manufacturing. Springer-Verlag, September, 2003, pp. 243 - 256.

E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V. Grunert da Fonseca. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. IEEE Transactions on Evolutionary Computation 7:2, April, 2003, pp. 117 - 132. (PDF, 1172 Kb)

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Transactions on Evolutionary Computation 6:2, April, 2002, pp. 182 - 197.

M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining Convergence and Diversity in Evolutionary Multi-objective Optimization. Evolutionary Computation 10:3, Fall, 2002, pp. 263 - 282.

K. Deb, L. Thiele, M. Laumanns, and E. Zitzler. Scalable Test Problems for Evolutionary Multiobjective Optimization. In A. Abraham, L. Jain, and R. Goldberg (Eds.), Evolutionary Multiobjective Optimization. Theoretical Advances and Applications. Springer, USA, 2005, pp. 105 - 145.