Algorytmy ewolucyjne

Piotr Lipiński

Klasyczny problem optymalizacji

- Niech $F : \Omega \to R$ będzie funkcją określoną na zbiorze Ω i przyjmującą wartości rzeczywiste.
- Problem minimalizacji funkcji F na zadanym zbiorze $D \subseteq \Omega$ polega na znalezieniu minimum funkcji F na zbiorze D (i punktu, w którym funkcja to minimum przyjmuje), tzn. na znalezieniu $x_0 \in D$ takiego, że dla każdego $x \in D \setminus \{x_0\}$ zachodzi $F(x_0) < F(x)$.
- Problem maksymalizacji funkcji F na zadanym zbiorze $D \subseteq \Omega$ polega na znalezieniu maksimum funkcji F na zbiorze D (i punktu, w którym funkcja to maksimum przyjmuje), tzn. na znalezieniu $x_0 \in D$ takiego, że dla każdego $x \in D \setminus \{x_0\}$ zachodzi $F(x) < F(x_0)$.
- Każdy problem minimalizacji może zostać przekształcony do problemu maksymalizacji i każdy problem maksymalizacji może zostać przekształcony do problemu minimalizacji.
- UWAGI:
  - Zbiór Ω nazywamy **przestrzenią poszukiwań**.
  - Funkcję F nazywamy **funkcją celu**.
  - Zbiór D nazywamy **zbiorem rozwiązań dopuszczalnych**.
  - W praktyce, często rozszerza się problem zastępując nierówności ostre nieostrymi i starając się znaleźć wszystkie punkty x_0 spełniające powstały warunek.
  - W praktyce, często nie jest ważne znalezienie dokładnych punktów minimów czy maksimów funkcji celu, a jedynie ich przybliżeń lub punktów, w których funkcja celu przyjmuje wartości "dostatecznie niskie" lub "dostatecznie wysokie" z praktycznego punktu widzenia.

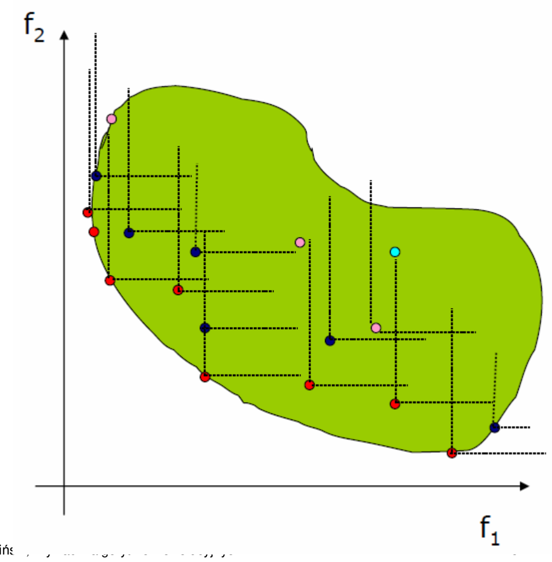
Klasyczny problem optymalizacji

- Przykład 1 (funkcje analityczne):
  - Znaleźć minimum funkcji $F(x) = 3x^2 - 2x + 1$.
  - Znaleźć minimum funkcji $F(x, y) = x^{17} y^{13} - 2x^{11} + 3y^7 + 5$.
  - Znaleźć minimum funkcji F(x), określonej jak wyżej, ale rozwiązaniami dopuszczalnymi są jedynie liczby całkowite.
- Przykład 2 (problemy kombinatoryczne):
  - Problem komiwojażera.
  - Problem QAP.
  - Problem Q3AP.
- Przykład 3 (funkcje "obliczeniowe" / Black Box Optimization):
  - W praktyce często mamy do czynienia z funkcjami, których dokładne matematyczne definicje i wzory nie są znane. Funkcje takie są obliczane w zadany sposób, na przykład w procesie symulacji pewnych zjawisk. Cała nasza wiedza o takiej funkcji to znajomość procedury obliczania jej wartości dla zadanego argumentu.
- Przykład 4:
  - Funkcje pochodzące z systemów uczenia maszynowego: systemów ekspertowych, systemów klasyfikujących, systemów wspomagania decyzji.
- Przykład 5:
  - Funkcje określone na drzewach, grafach czy innych złożonych obiektach.

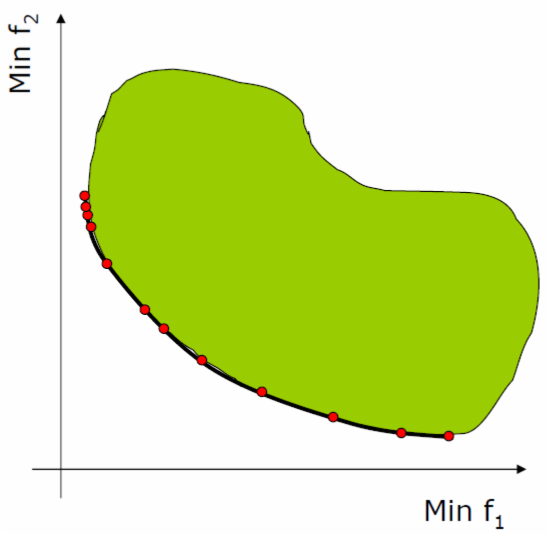
Problem optymalizacji wielokryterialnej

- Niech Ω będzie przestrzenią poszukiwań.
- Rozpatrujemy K funkcji celu: $F_1 : \Omega \to R, F_2 : \Omega \to R, \ldots, F_K : \Omega \to R$.
  - Funkcje te definiują różne kryteria oceny elementów przestrzeni poszukiwań. Zwykle nazywa się je funkcjami kryteriów, a nie funkcjami celu. Stąd też pochodzi nazwa problemu – optymalizacja wielokryterialna.
  - Przyjmijmy, że problem polega na minimalizacji funkcji kryteriów. Problem, w którym niektóre funkcje kryteriów należy maksymalizować można sprowadzić do równoważnego problemu z minimalizowaniem wszystkich funkcji kryteriów.
  - Funkcje kryteriów zazwyczaj są ze sobą sprzeczne (maksymalizacja jednej funkcji pociąga za sobą minimalizację innej).
- Element $\mathbf{x} \in \Omega$ dominuje element $\mathbf{y} \in \Omega$, jeśli $F_k(\mathbf{x}) \le F_k(\mathbf{y})$ dla każdego k = 1, 2, …, K oraz $F_k(\mathbf{x}) < F_k(\mathbf{y})$ dla pewnego k = 1, 2, …, K.
  - Element $\mathbf{y} \in \Omega$ jest elementem zdominowanym jeśli istnieje element $\mathbf{x} \in \Omega$, który go dominuje.
  - Element $\mathbf{y} \in \Omega$ jest elementem niezdominowanym jeśli nie istnieje element $x \in \Omega$, który go dominuje.
- Zbiór elementów niezdominowanych nazywamy frontem Pareto.
- Problem optymalizacji wielokryterialnej z przestrzenią poszukiwań Ω i funkcjami kryteriów $F_1 : \Omega \to R, F_2 : \Omega \to R, \ldots, F_K : \Omega \to R$ polega na wyznaczeniu frontu Pareto.
- Analogicznie określa się problem optymalizacji wielokryterialnej z ograniczeniami.

Problem optymalizacji wielokryterialnej



Problem optymalizacji wielokryterialnej

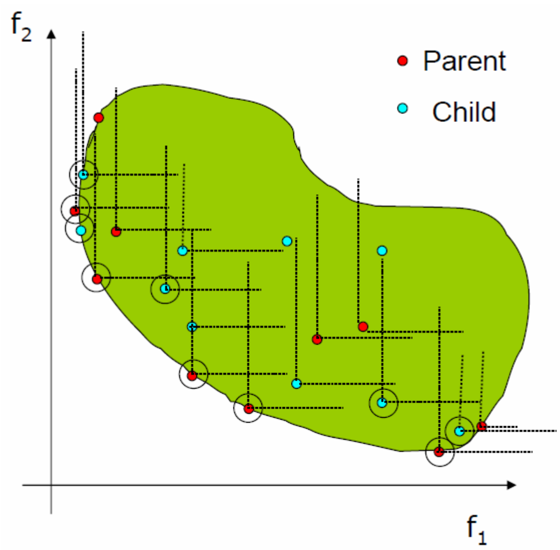


Problem optymalizacji wielokryterialnej

- Popularne algorytmy ewolucyjne do optymalizacji wielokryterialnej:
  - Multi-Objective Genetic Algorithm, MOGA (Fonseca, Fleming, 1993)
  - Non-Dominated Sorting Genetic Algorithm, NSGA (Srinivas, Deb, 1995)
  - Niched-Pareto Genetic Algorithm, NPGA (Horn, Nafpliotis, Goldberg, 1994)
  - Non-Dominated Sorting Genetic Algorithm II, NSGA-II (Deb, et al.., 2000)
  - SPEA, SPEA2
  - MOEA/D

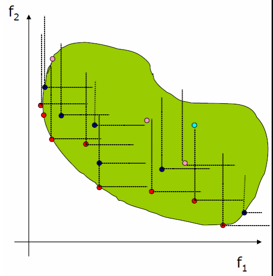
NSGA-II

- Oryginalny algorytm NSGA-II opiera się na SGA, w którym zmieniono operator wyboru nowej populacji.



NSGA-II – Fast Nondominated Sorting

- Jak wyznaczać kolejne fronty Pareto?
  - Niech M oznacza liczbę kryteriów, a N oznacza liczbę osobników.
- Naiwnie - $O(MN^3)$:
  - potrzeba O(MN) porównań do określenia czy dany osobnik jest zdominowany czy niezdominowany
  - potrzeba $O(MN^2)$ porównań do wyznaczenie frontu Pareto (PF0)
  - potrzeba $O(MN^3)$ porównań do wyznaczenie wszystkich kolejnych frontów Pareto (PF0, PF1, …)
    - przypadek pesymistyczny: może być N jednoelementowych frontów Pareto

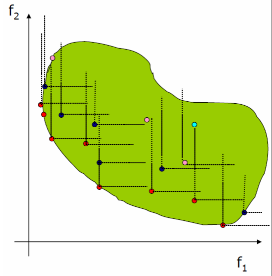


NSGA-II – Fast Nondominated Sorting

- Jak wyznaczać kolejne fronty Pareto?
  - Niech M oznacza liczbę kryteriów, a N oznacza liczbę osobników.
- Sprytnie - $O(MN^2)$:
  - Dla każdego osobnika **x**, wyznaczamy *domination count* N**x** (liczbę osobników, które dominują **x**) oraz zbiór S**x** (zbiór osobników, które dominuje osobnik **x**). Wymaga to $O(MN^2)$ porównań.
  - Osobniki z PF0 będą miały N**x** = 0.
  - Dla każdego osobnika **x**, który ma N**x** = 0 odwiedzamy każdego osobnika **y** z S**x** i obniżamy jego N**y** o 1. Jeśli po obniżeniu N**y** wyniesie 0, to umieszczamy osobnika na pomocniczej liście Q. Osobniki, które znajdą się na liście Q tworzą PF1.
  - Powtarzamy poprzedni punkt dla osobników z listy Q, dopóki jest ona niepusta (czyszcząc listę Q i tworząc ją od nowa). Osobniki, które w kolejnych iteracjach znajdą się na liście Q będą tworzyć PF2, PF3, ….

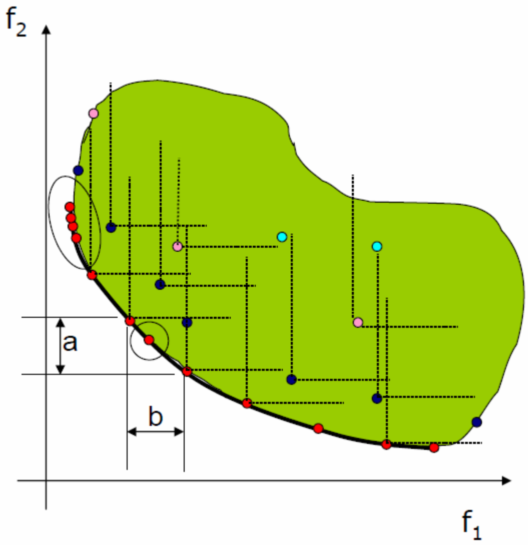
NSGA-II – Fast Nondominated Sorting

- Jak wyznaczać kolejne fronty Pareto?
  - Niech M oznacza liczbę kryteriów, a N oznacza liczbę osobników.
- Sprytnie - $O(MN^2)$:
  - Dla każdego osobnika **y** z PF1, PF2 i kolejnych, początkowa N**y** może wynosić najwyżej N-1 (bo N to liczba wszystkich osobników, a co najmniej jeden z nich musi być w PF0).
  - Zatem każdy osobnik może być odwiedzony co najwyżej N-1 razy zanim jego N**y** spadnie do zera.
  - Zatem wyznaczenie PF1, PF2, i dalszych ma złożoność $O(N^2)$.
  - Łączna złożoność więc to $O(MN^2 + N^2) = O(MN^2)$.



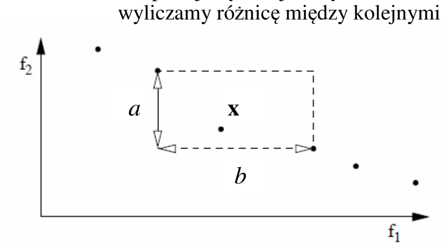
NSGA-II – Crowding Distance

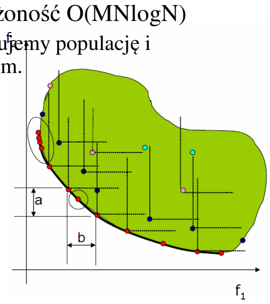
- Jak wymusić równomierne rozproszenie frontu Pareto?



NSGA-II – Crowding Distance

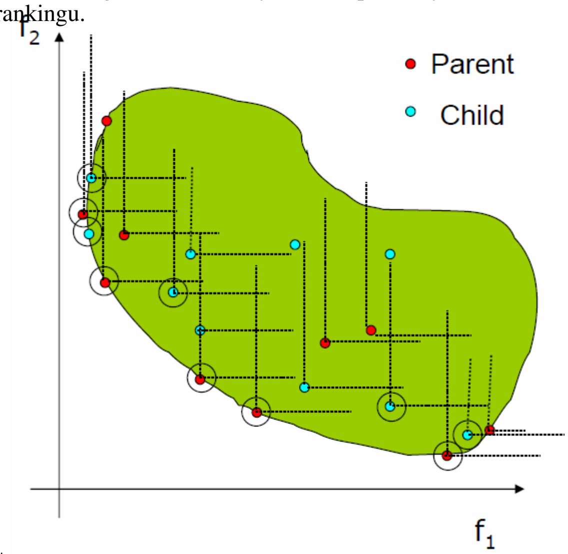
- W NSGA-II wprowadzono miarę oceny osobników zwaną Crowding Distance:
  - załóżmy, że rozpatrujemy problem optymalizacji dwukryterialnej
  - dla osobnika **x** znajdujemy w aktualnej populacji jego "dwóch najbliższych sąsiadów" **y** i **z** (w przestrzeni kryteriów, nie w przestrzeni poszukiwań!):
    - pierwszy sąsiad **y** ma być lepszy od **x** względem pierwszego kryterium
    - drugi sąsiad **z** ma być lepszy od **x** względem drugiego kryterium
  - punkty **y** i **z** rozpinają prostokąt, długości boków tego prostokąta oznaczmy przez *a* i *b*
  - Crowding Distance osobnika **x**, to CR(**x**) = *a* + *b*.
  - jeśli jeden z sąsiadów nie istnieje, to przyjmujemy, że odpowiedni bok prostokąta ma długość nieskończoną, czyli CR(**x**) = ∞.
- Policzenie Crowding Distance dla całej populacji ma złożoność O(MNlogN)
  - Rozpatrujemy kolejne kryteria, dla każdego kryterium sortujemy populację i wyliczamy różnicę między kolejnymi wartościami kryterium.





NSGA-II – Ranking osobników

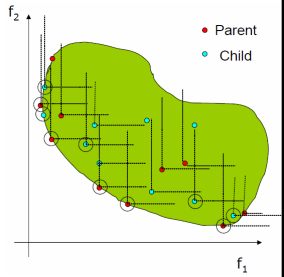
- Fast Nondominated Sorting i Crowding Distance są używane do porównywania osobników i tworzenia ich rankingu.



NSGA-II – Ranking osobników

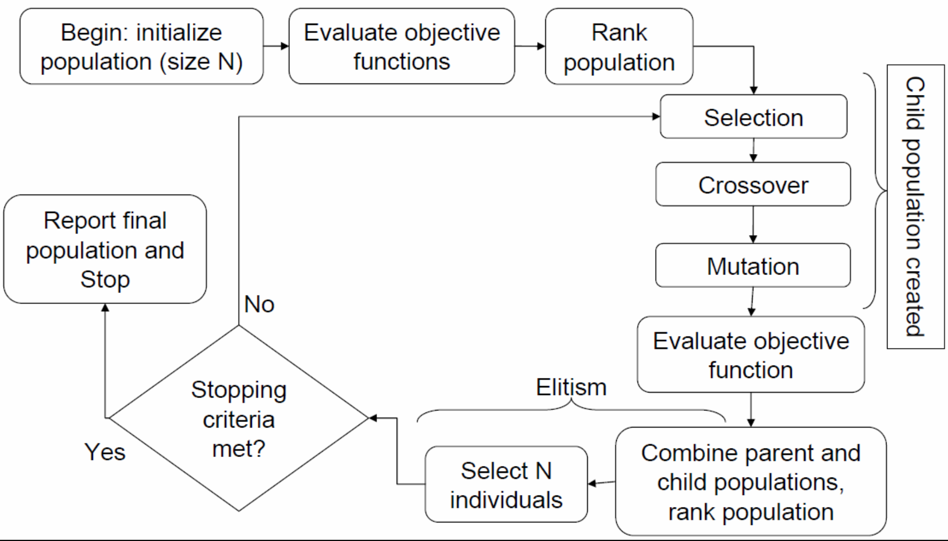
- Ranking tworzony jest według dwóch kryteriów:
  - ranga osobnika (czyli numer frontu Pareto do którego należy, im niższy tym lepiej)
  - Crowding Distance osobnika (im wyższy tym lepiej)

- Dokładniej: Definujemy relację < na zbiorze osobników. Niech **x** i **y** będą dwoma osobnikami, wówczas:
  - Jeśli rang(**x**) < rang(**y**), to **x** < **y**
  - Jeśli rang(**x**) > rang(**y**), to **y** < **x**
  - Jeśli rang(**x**) = rang(**y**), to
    - Jeśli CR(**x**) > CR(**y**), to **x** < **y**
    - Jeśli CR(**x**) < CR(**y**), to **y** < **x**
    - Jeśli CR(**x**) = CR(**y**), to **dowolnie**



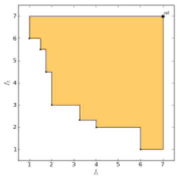
NSGA-II

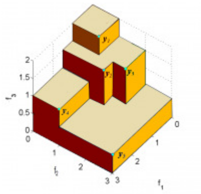
- Oryginalny algorytm NSGA-II opiera się na SGA, w którym zmieniono operator wyboru nowej populacji.



Hypervolume

- Jak oceniać działanie algorytmu optymalizacji wielokryterialnej?
- Celem algorytmu jest wyznaczenie frontu Pareto, a dokładniej:
  - umieszczenie osobników na dokładnym froncie Pareto,
  - umieszczenie ich w taki sposób, żeby były "równomiernie rozproszone".
- Popularna używana miara jakości to hypervolume (HV)





- Jeśli znany jest dokładny front Pareto FP*, to można policzyć relative hypervolume (HVR) jako HVR(FP) = HVR(FP) / HVR(FP*).

MOO dla problemów optymalizacji z ograniczeniami

- Infeasibility Driven Evolutionary Algorithm (IDEA) to przykład algorytmu opartego na NSGA-II do rozwiązywania problemów optymalizacji z ograniczeniami.
  - Pierwsze kryterium $F_1 = F: \Omega \rightarrow R$ definiowane jest przez oryginalną funkcję celu.
  - Drugie kryterium $F_2 : \Omega \rightarrow R$ definiowane jest przez "stopień złamania ograniczeń".