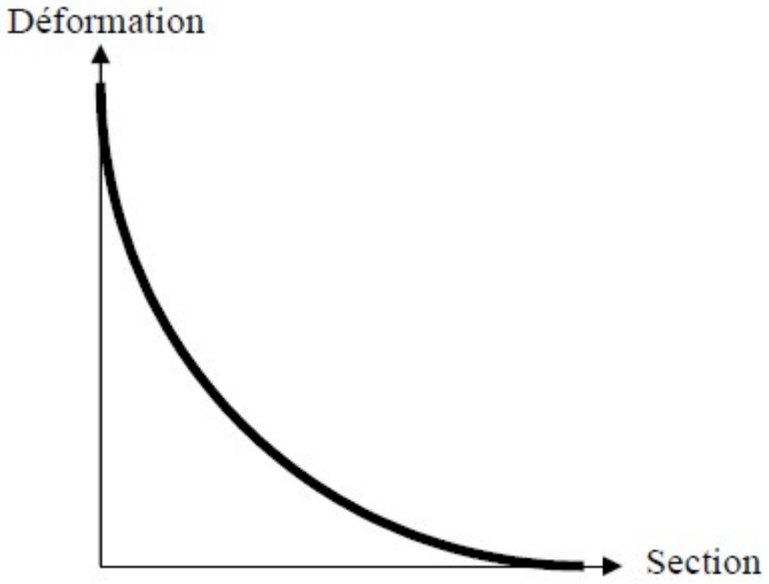
Algorithmique avancée
Méthodes computationnelles

Laurent Lemarchand
Lab-STICC/UBO
Laurent.Lemarchand@univ-brest.fr

Optimisation multi objectifs (MOO)

Optimisation combinatoire Multi-Objectifs
introduction à la MOO

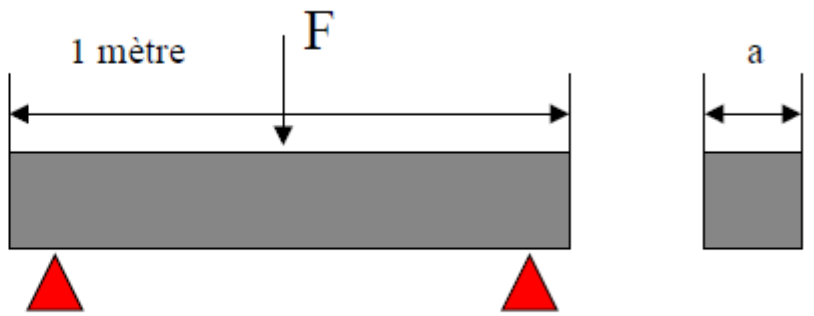


- Méthodes d'optimisation (suivant objectifs qualitatifs et critères simples ou **multiples**)
- La poutre : **section** (poids) VS **déformation**

$$S(a) = a^2$$

$$d(a) = 1000 + \frac{1.10^{-2}}{192 + 2.10^{5} + \frac{a^4}{12}}$$

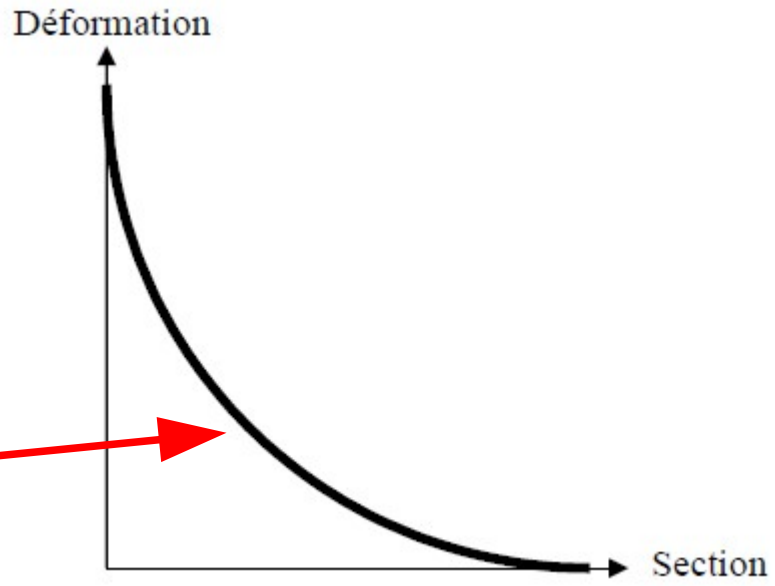
$$a \leq 0.1$$



Optimisation combinatoire
introduction à la MOO

≠ Mono-objectif

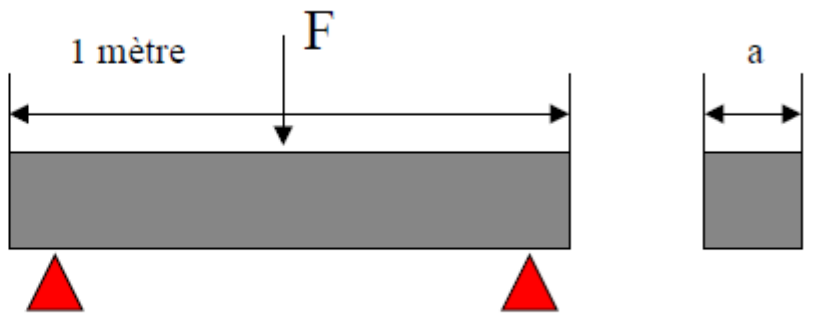
- **Plusieurs** fonctions objectives
  - antagonisme
- Pas de *meilleure* solution
  - **ensemble** de solutions



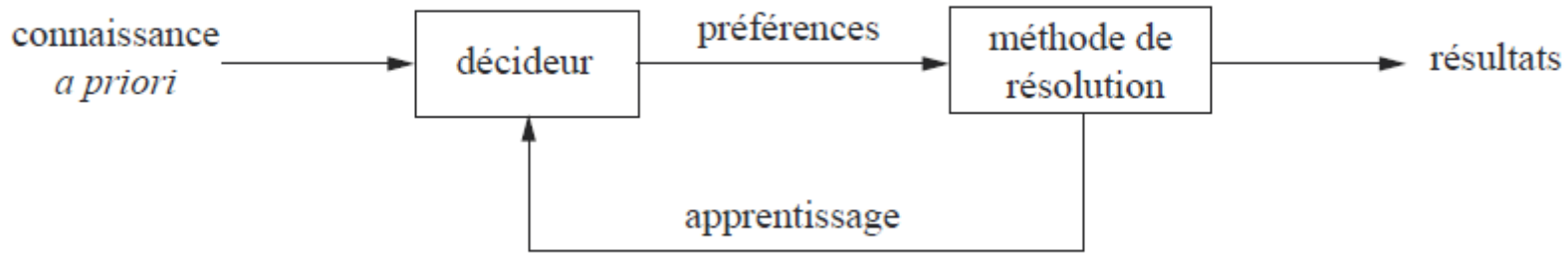
$$S(a) = a^2$$

$$d(a) = 1000 + \frac{1.10^{-2}}{192 + 2.10^5 + \frac{a^4}{12}}$$

$$a \leq 0.1$$



Recherche des solutions
décideur



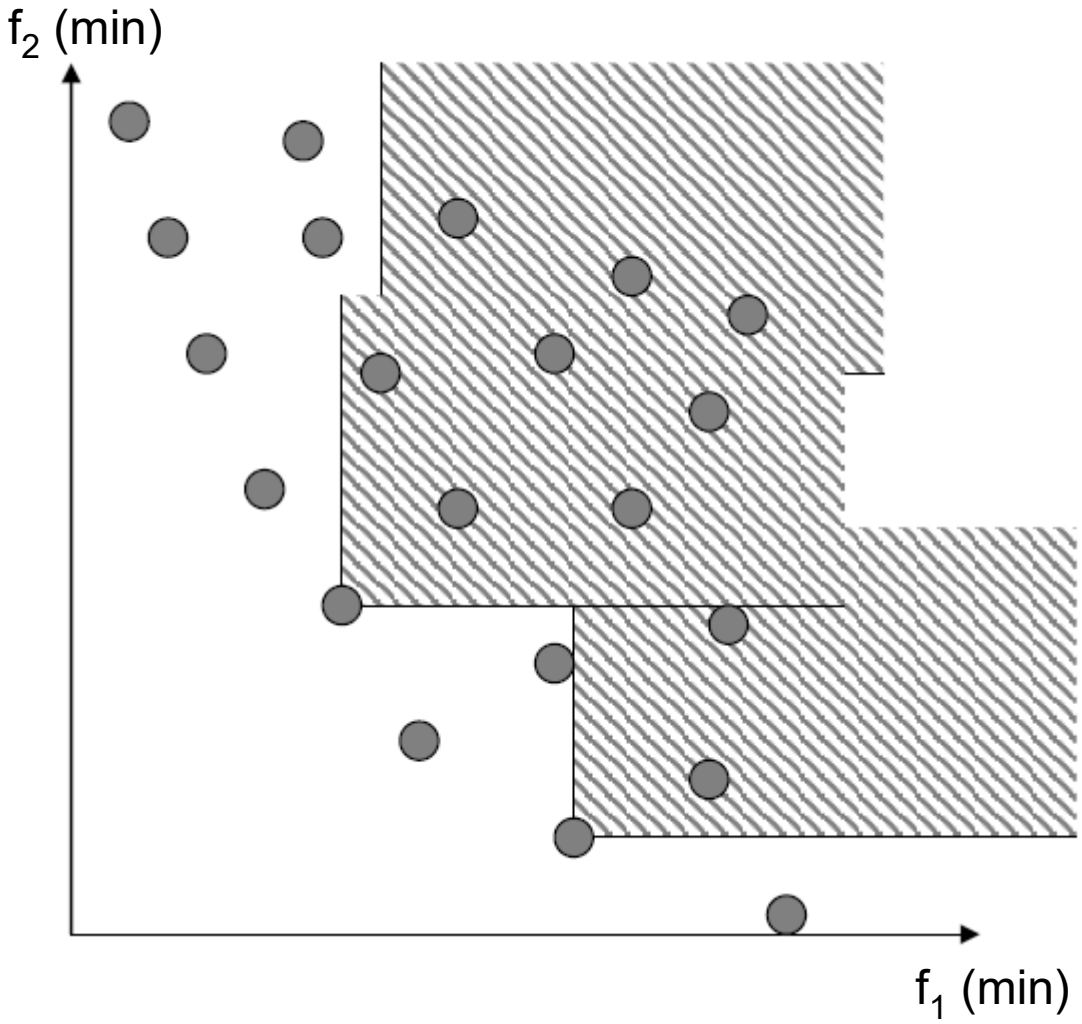
- Recherche a priori
  - Priorités (ex. méthode de compromis)
- Recherche a posteriori → tout l'ensemble des solutions
  - Peut être complexe à analyser
- Recherche interactive
  - Guidage du décideur

Introduction à la MOO
Dominance

- Comment comparer les solutions entre elles ?
- Une solution a **domine** une solution b si
  - a est aussi bonne que b sur tous les critères i d'optimisation:

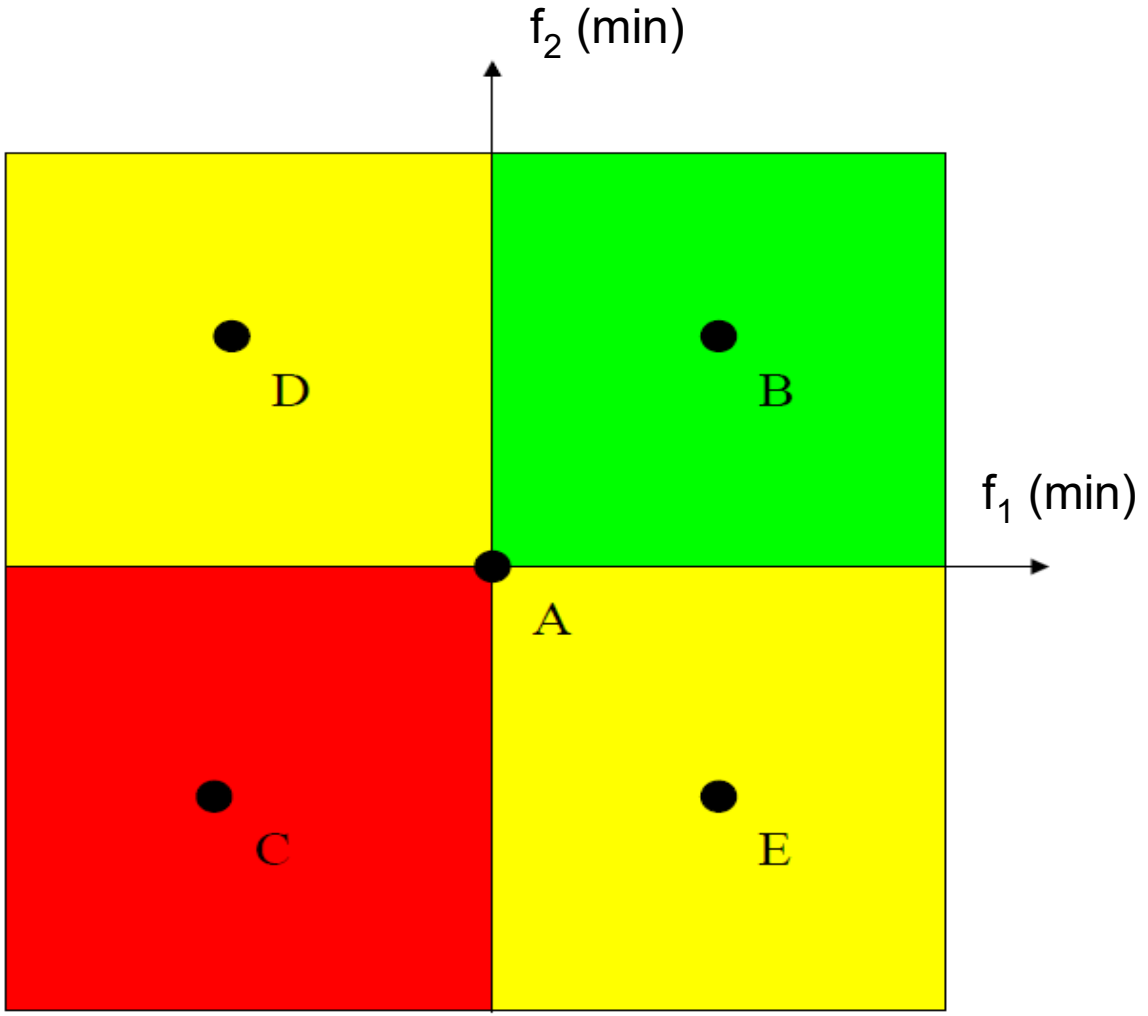
    $\forall$ i, $f_i(a) \leq f_i(b)$
  - Il y a au moins un critère j pour lequel a est meilleure que b :

    $\exists$ $f_j(a) < f_j(b)$



Introduction à la MOO
Dominance

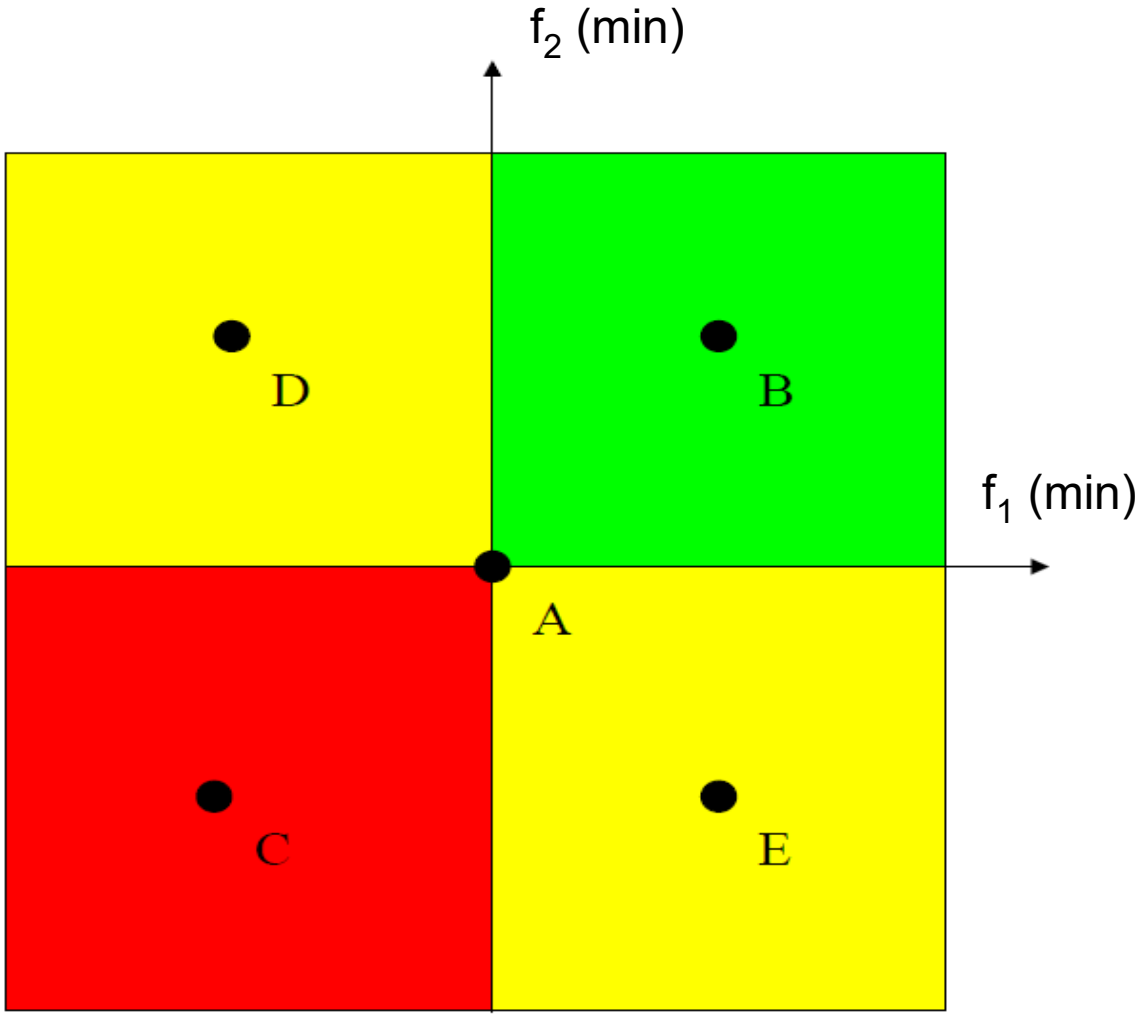
- Qui est dominé par A ?
- Qui domine A ?
- Qui est non comparable à A ?



Introduction à la MOO
Dominance

- Qui est dominé par A ?
  B
- Qui domine A ?
  C
- Qui est non comparable à A ?

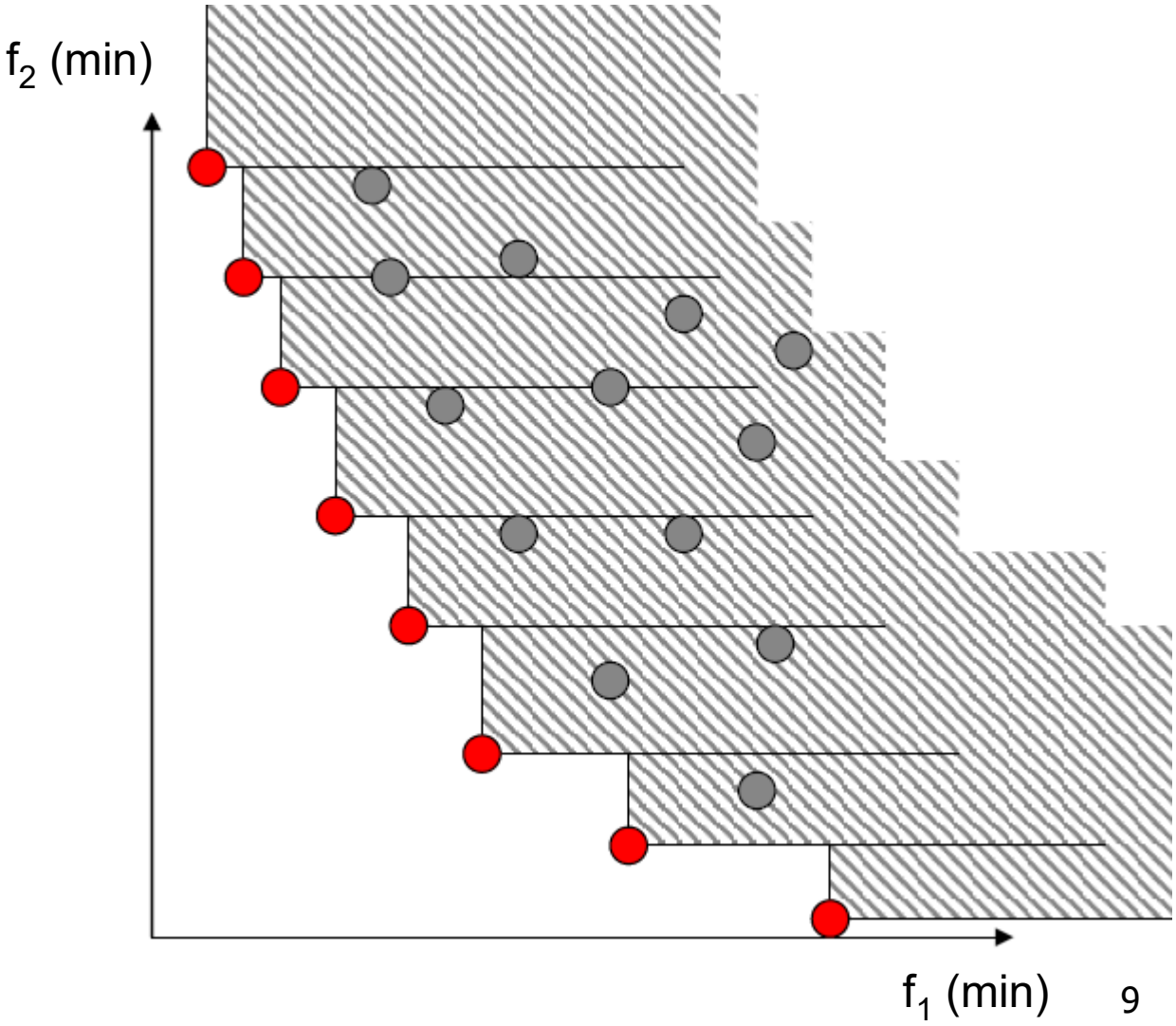
  D et E



Introduction à la MOO
Front de Pareto (1/2)

- Ensemble des solutions **non-dominées**
  - Solutions optimales au sens de Pareto
  - Front de Pareto maximal/minimal : toutes/une seule des solutions pour un même avec vecteur objectif identique

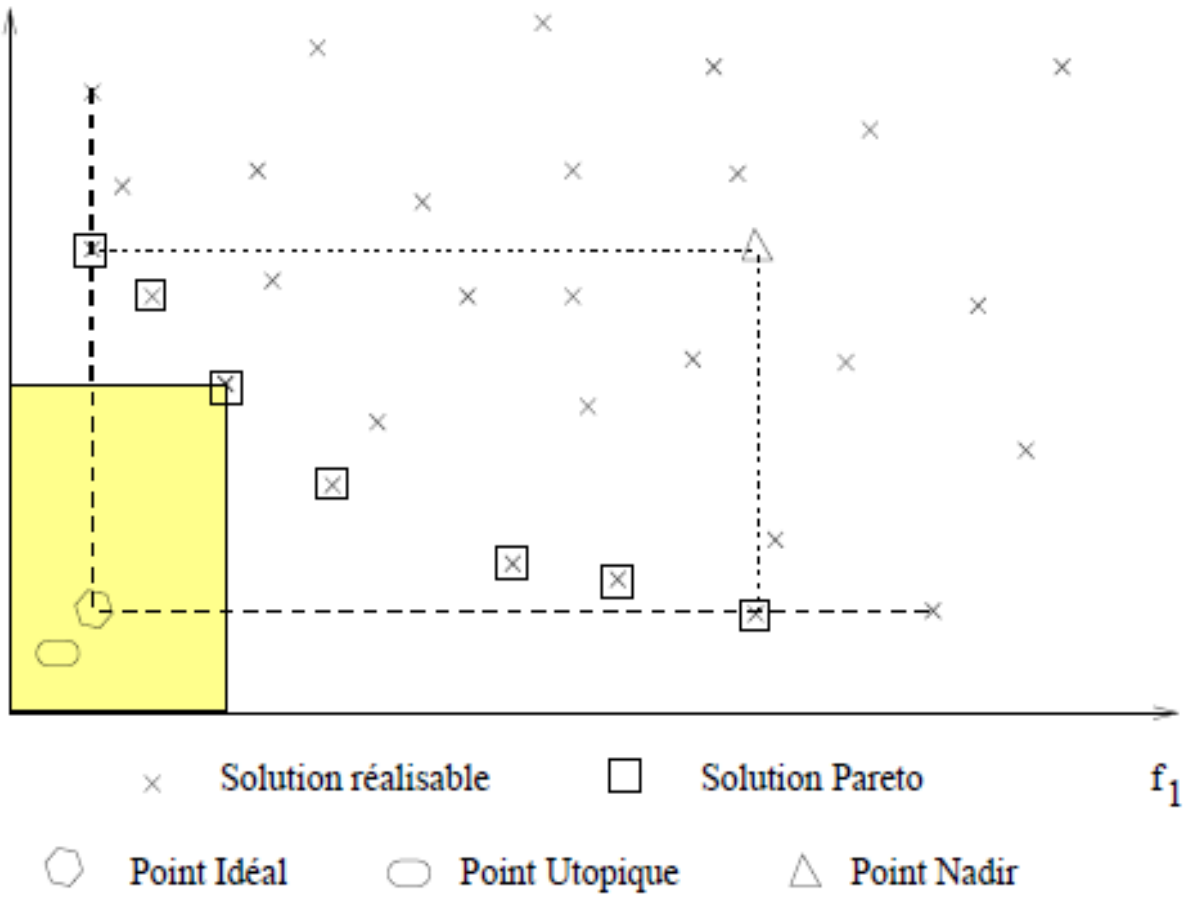
les algorithmes MOO cherchent en général un front min.



D'après Y. Collette – Renault Technocentre

Introduction à la MOO
Front de Pareto (2/2)

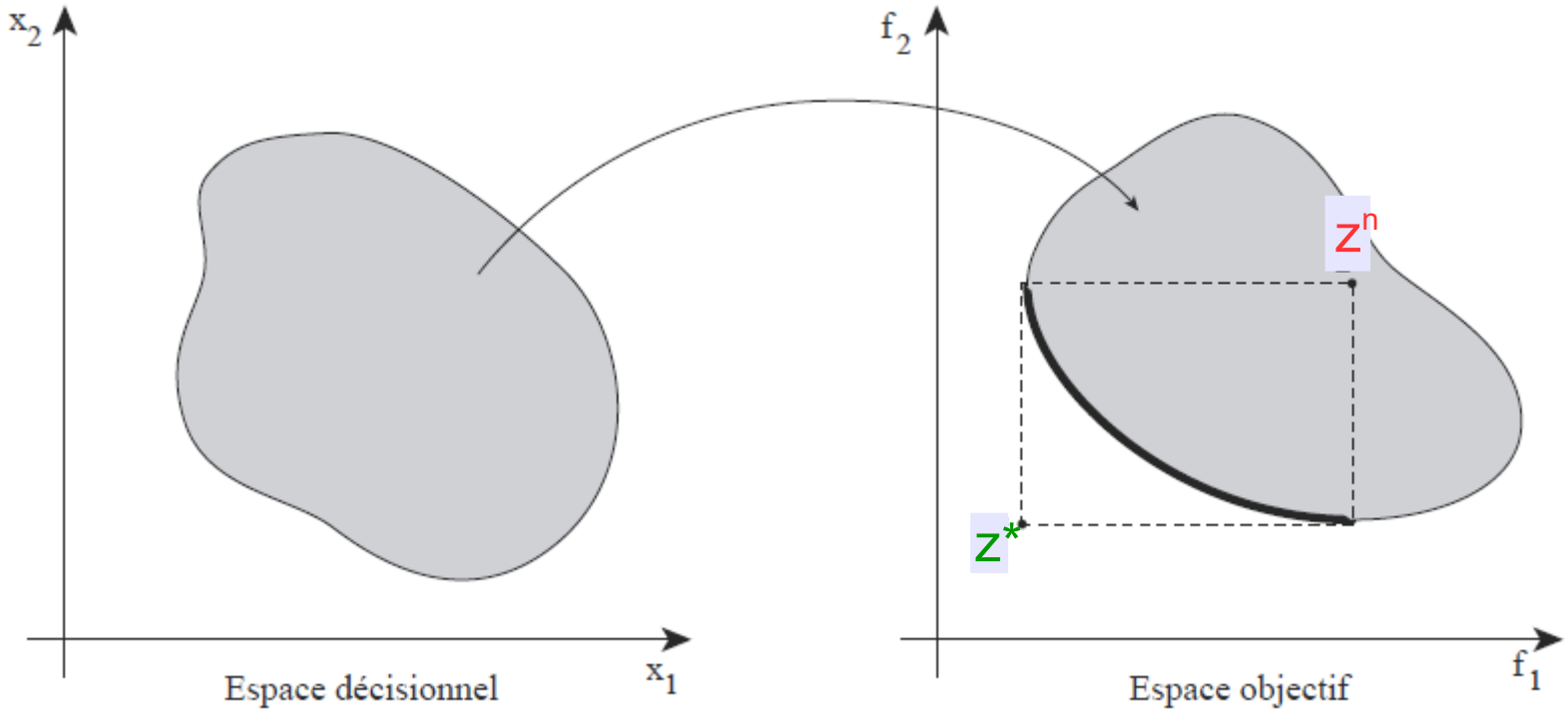
- **Points remarquables**
  - Point idéal
  - Point utopique
  - Points de Pareto
  - Solutions réalisables
  - Point Nadir
  - Bornes pas forcement calculables



D'après Clarisse Dhaenens-Flipo

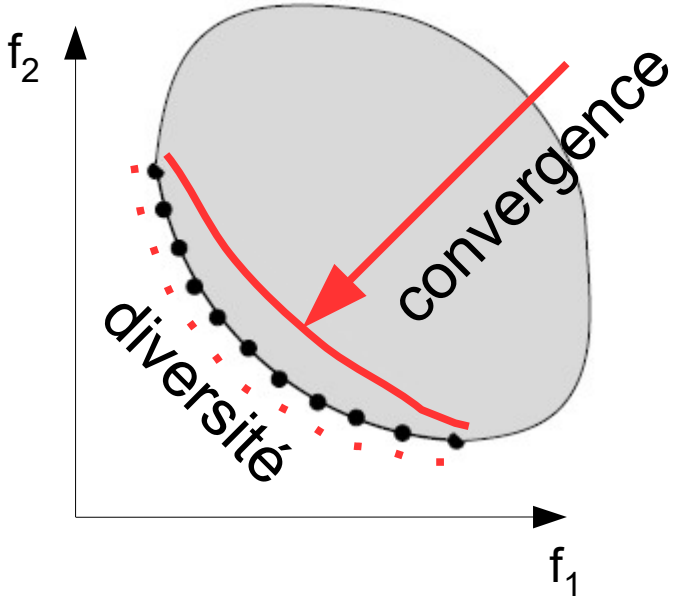
Qualité des solutions
propriétés

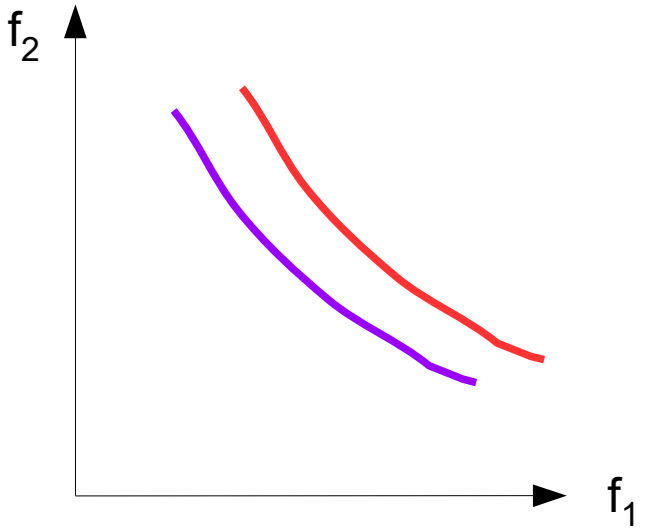
- Rappel : points de référence z^n (point Nadir) et z^* (point idéal) dans l'espace objectif
  - Bornes, pas forcement connues ou calculables

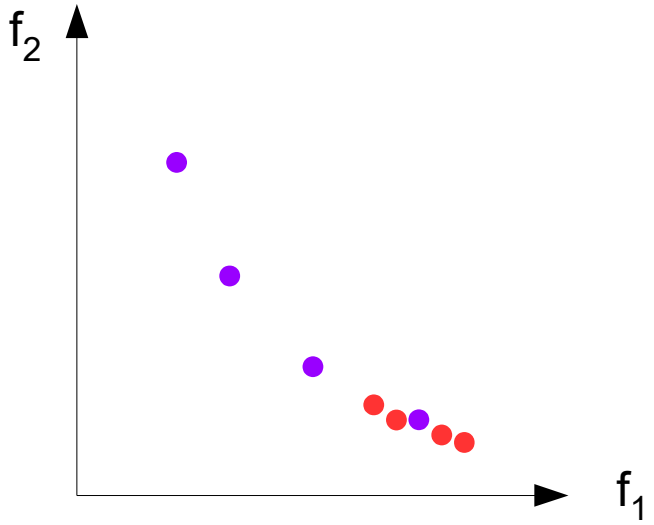


Qualité des solutions
mesures comparatives

- Convergence
  - s'approcher du front de Pareto
- Diversité
  - Garantir la répartition
  - Obtenir un nombre de solutions suffisant







Qualité des solutions
mesures comparatives

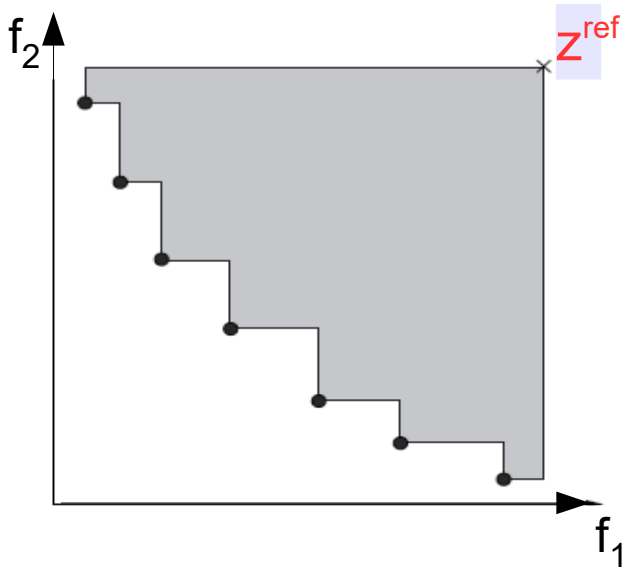
- Indicateur I Pareto-conforme ou non
  - Si tous les élements d'un ensemble A sont dominés (ou égaux) par un élément d'un ensemble B, I(B) meilleur que I(A)
  - GD (distance au front P) non conforme
  - H (surface de Lebesgue, hypervolume) conforme

- Comparaison au vrai front de Pareto (mesure unaire) ou entre fronts (mesure binaire)
  - Ex : ER (**ratio d'erreurs**) d'un front F par rapport au front réel P

$$\frac{|\{ f \in F \text{ t.q } f \notin P \}|}{|P|}$$

Qualité des solutions
mesures comparatives

- Mesure unitaire Pareto conforme
  ex : **hyper volume**
  - Par rapport à un point de référence
  - Normalisation des fcts objectives

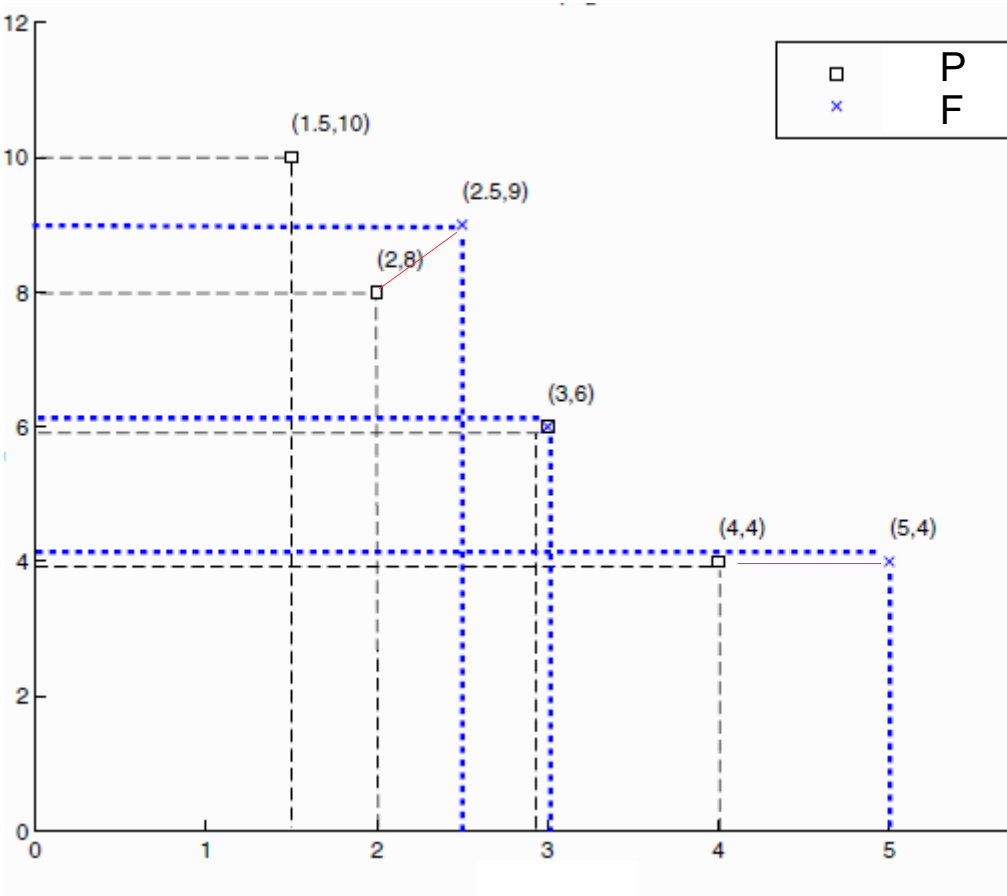


  - z^{ref} obtenu comme étant le point Nadir de l'ensemble des essais/algorithmes à comparer

Qualité des solutions
mesures comparatives

- **Distance générationnelle** GD unitaire non Pareto conforme
  - Par rapport au front de Pareto
  - Distance moyenne pour chaque point i de F au point le plus proche dans P (p = #fcts obj.)

$$GD \triangleq \frac{\left(\sum_{i=1}^{n} d_i^p\right)^{1/p}}{|P|}$$



ex : $d_1 = \sqrt{(2.5-2)^2 + (9-8)^2}$

$d_2 = \sqrt{(3-3)^2 + (6-6)^2}$

$d_3 = \sqrt{(5-4)^2 + (4-4)^2}$

GD = $\sqrt{1.118^2 + 0^2 + 1^2}$ / 3 = 0.5

Qualité des solutions
mesures comparatives

- **Distance générationnelle inverse** IGD toujours par rapport au front de Pareto
  - Moyenne des distances pour chaque point i de P* au point le plus proche dans A

$$IGD(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|}$$

Qualité des solutions
mesures comparatives

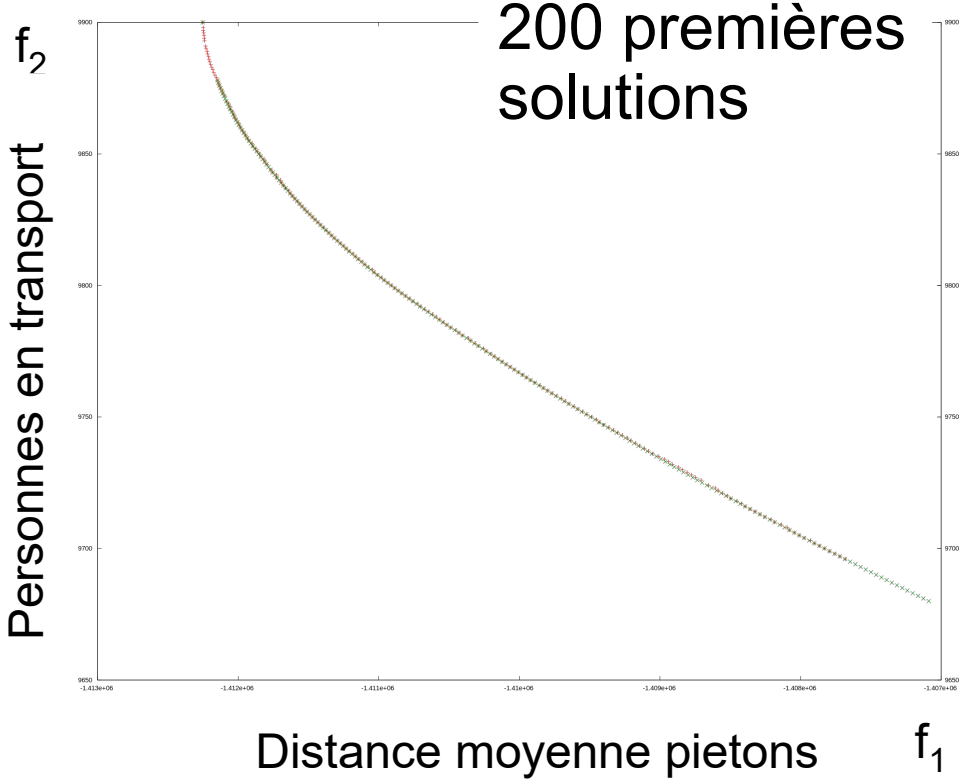
- Problème de la mise à l'échelle
  - Si normalisation mal faite
  - Privilégier implicitement un objectif

- Ex : conversion kilos → tonnes pour un objectif → dominance inversée

	f_1		f_2		$80\%f_1 + 20\%f_2$	
	Coût	**Déchets**		**Somme**		**Rang**	
a	100 000	5000	5 000 000	81 000	1 080 000	3	1
b	80 000	10000	10 000 000	66 000	2 064 000	2	2
c	40 000	20000	20 000 000	36 000	4 032 000	1	3

Introduction à la MOO
Algorithmes MOO

- Résolution **exacte**
  - Sur formulation IP (MOIP)
    - Méthode 2 phases
    - Méthode basée sur les ε-Contraintes
- Souvent limité
  - IP $\rightarrow$ NP-complet
  - MOIP $\rightarrow$ nombreux sous problèmes IP à résoudre
  - Ex : p-median MOO



Introduction à la MOO
Algorithmes MOO

- Résolution **approchée (ou partielle)**
  - Avec une technique mono-objectif
    - Agrégation
    - ε-Contraintes
    - But à atteindre
  - Méthodes non Pareto
    - VEGA : traitement indépendant des objectifs
    - Méthode lexicographique
  - Méthodes évolutionnaires MOO
    - Mono solutions (PESA, MOSA, PAES)
    - Populations (MOGA, NPGA 1994, NSGA-II 2002, …)
    - Méthodes basées sur les indicateurs (SMS-EMOA)

Techniques SOO pour MOO
agrégation (pondération des objectifs)

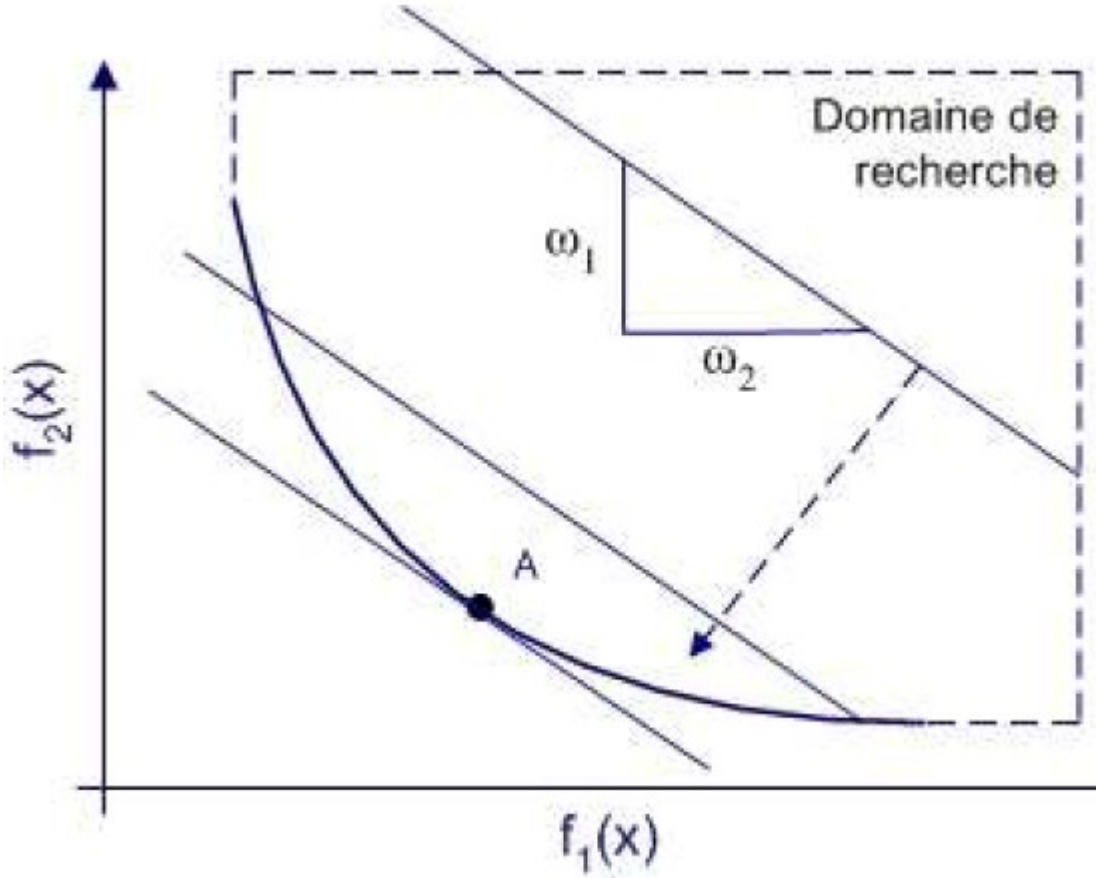
- *min* f(x) = ($f_1(x)$, $f_2(x)$, ..., $f_n(x)$)

  min f'(x) = $\omega_1.f_1(x) + \omega_2.f_2(x) + ... + \omega_n.f_n(x)$)

Combiner les objectifs

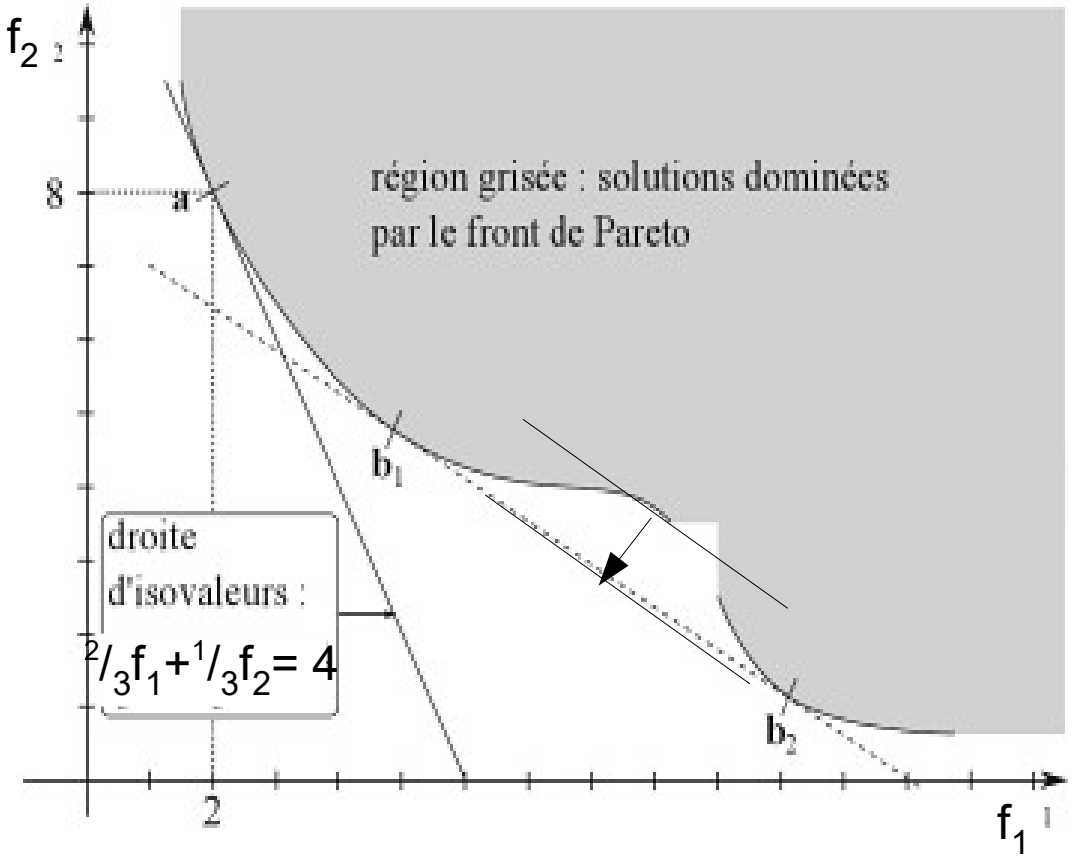
avec
$\omega_1 + ... + \omega_n = 1$

- Si espace convexe : point optimum A à la tangente de la droite de pente – ω_1 / ω_2



Techniques SOO pour MOO
agrégation (pondération des objectifs)

- Problème pour les fronts non convexes
  - Pas de combinaison ω_i possible pour certains points du front de Pareto soit le minimum pour f (*solutions non supportées*)
  - pour n'importe quelle droite essayant de passer par un des points entre b_1 et b_2, on peut encore la décaler pour avoir mieux



Techniques SOO pour MOO

ε-contraintes (compromis)

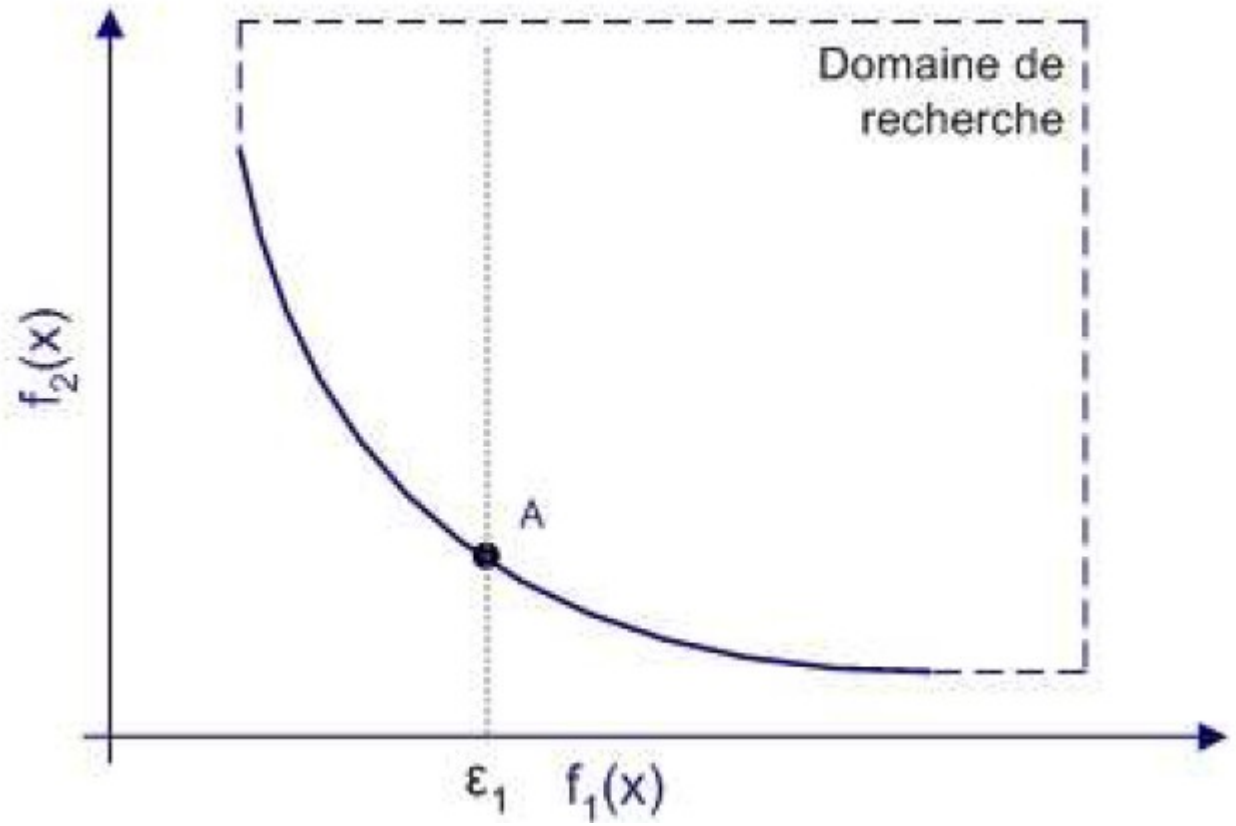
- *min* $f(x) = (f_1(x), f_2(x), ..., f_n(x))$

Choisir 1 objectif *j*
Contraintes pour les autres

$$\begin{cases} \textit{min}\ f_j(x) \\ f_i(x) \leq \varepsilon_i \quad \forall\ i \neq j \end{cases}$$

avec j et les ε_i choisis par l'utilisateur

- Si espace convexe : point optimum A à l'intersection du front et de la droite $f_1 = \varepsilon_1$



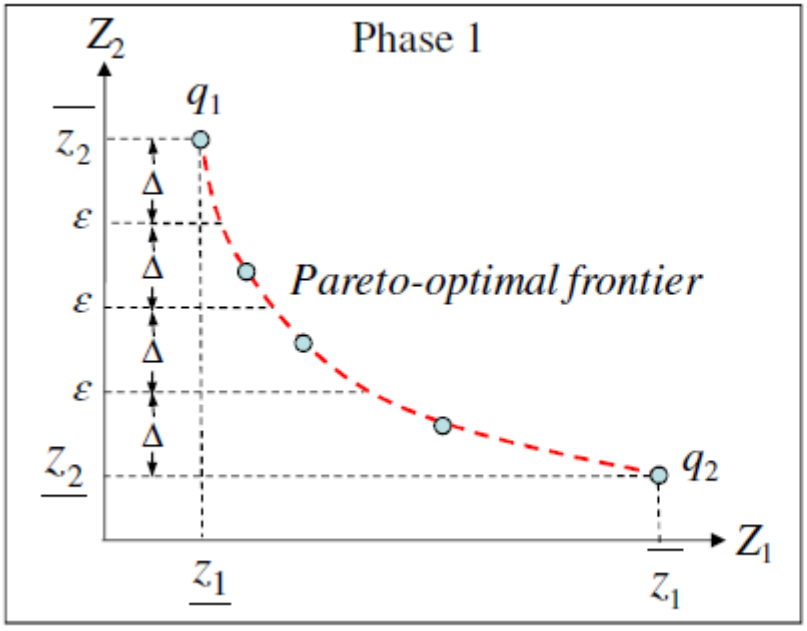
Techniques SOO pour MOO

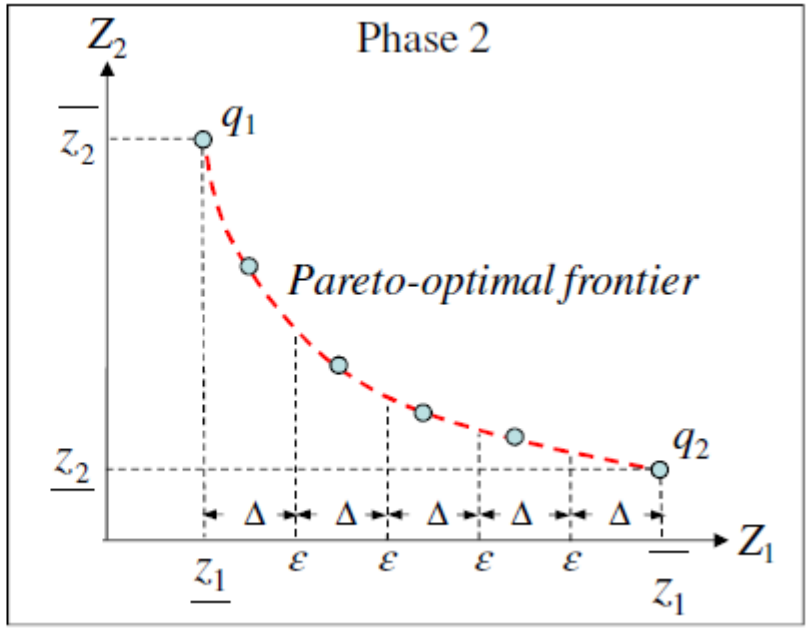
ε-contraintes (compromis)

- Exemple de génération de ε $\rightarrow$ algorithme 2 phases :
  - Calcul des extrema $[\underline{z}_1 .. \bar{z}_1]$ et $[\underline{z}_2 .. \bar{z}_2]$

    [en enlevant la seconde fonction objective]
  - Choix d'un intervalle de largeur Δ sur f_1 pour obtenir *s* points du front de Pareto

    [en ajoutant des contraintes d'intervalles]





Techniques SOO pour MOO
But à atteindre

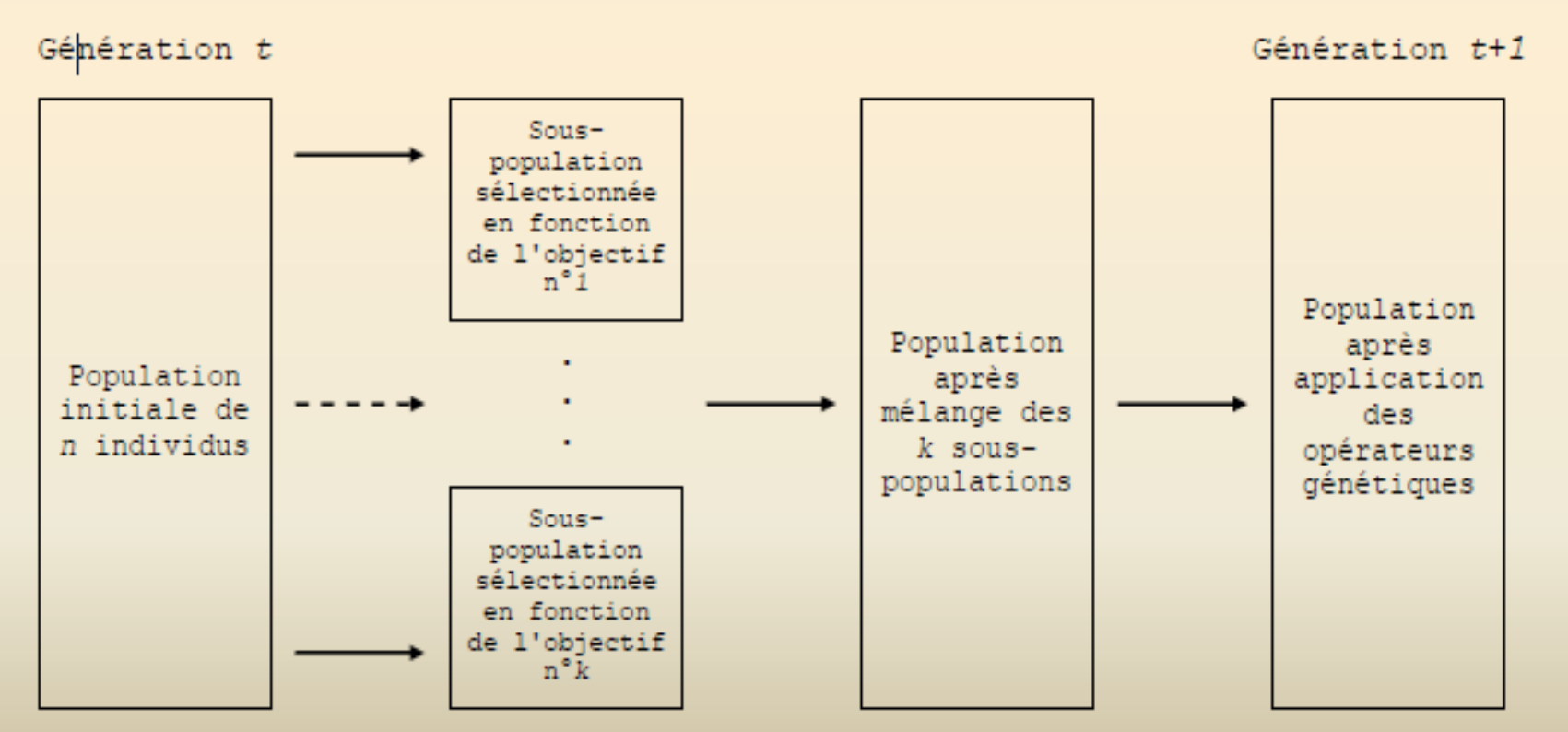
- *min* $f(x) = (f_1(x), f_2(x), \ldots, f_n(x))$

  min $f'(x) = |f_1(x) - T_1| + |f_2(x) - T_2| + \ldots + |f_n(x) - T_n|$

  - T_1, T_2, T_n valeurs à atteindre pour chaque objectif
  - On peut aussi pondérer chacun des objectifs (cf agrégation)

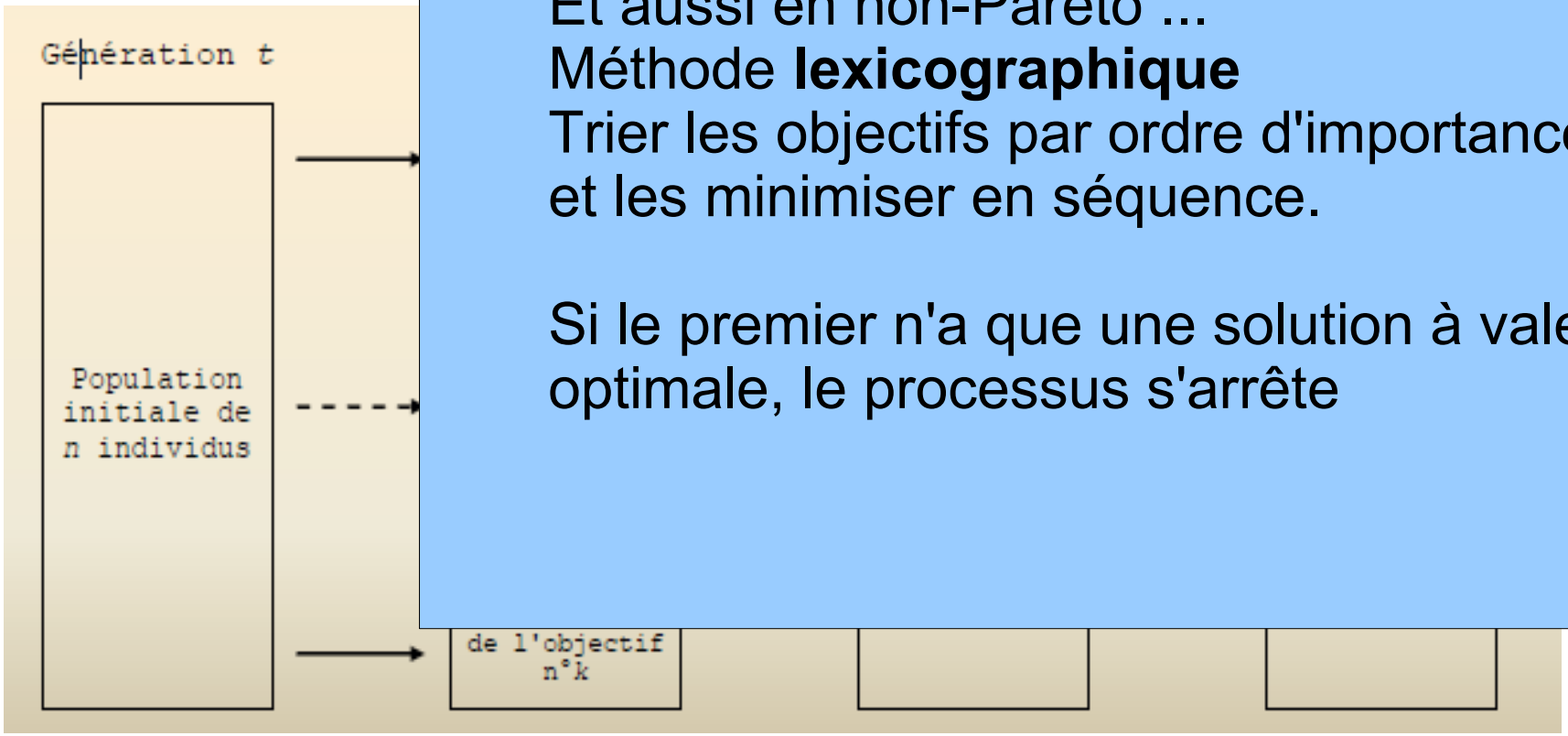
MOEA non Pareto
VEGA

- Vector Evaluated GA : ne base pas la sélection sur la dominance
- La seule différence avec un algorithme génétique est la manière dont s'effectue la **sélection**, n individus et k objectifs



MOEA non Pareto
VEGA

- Vector Evaluated G
- La seule différence
  s'effectue la **sélect**



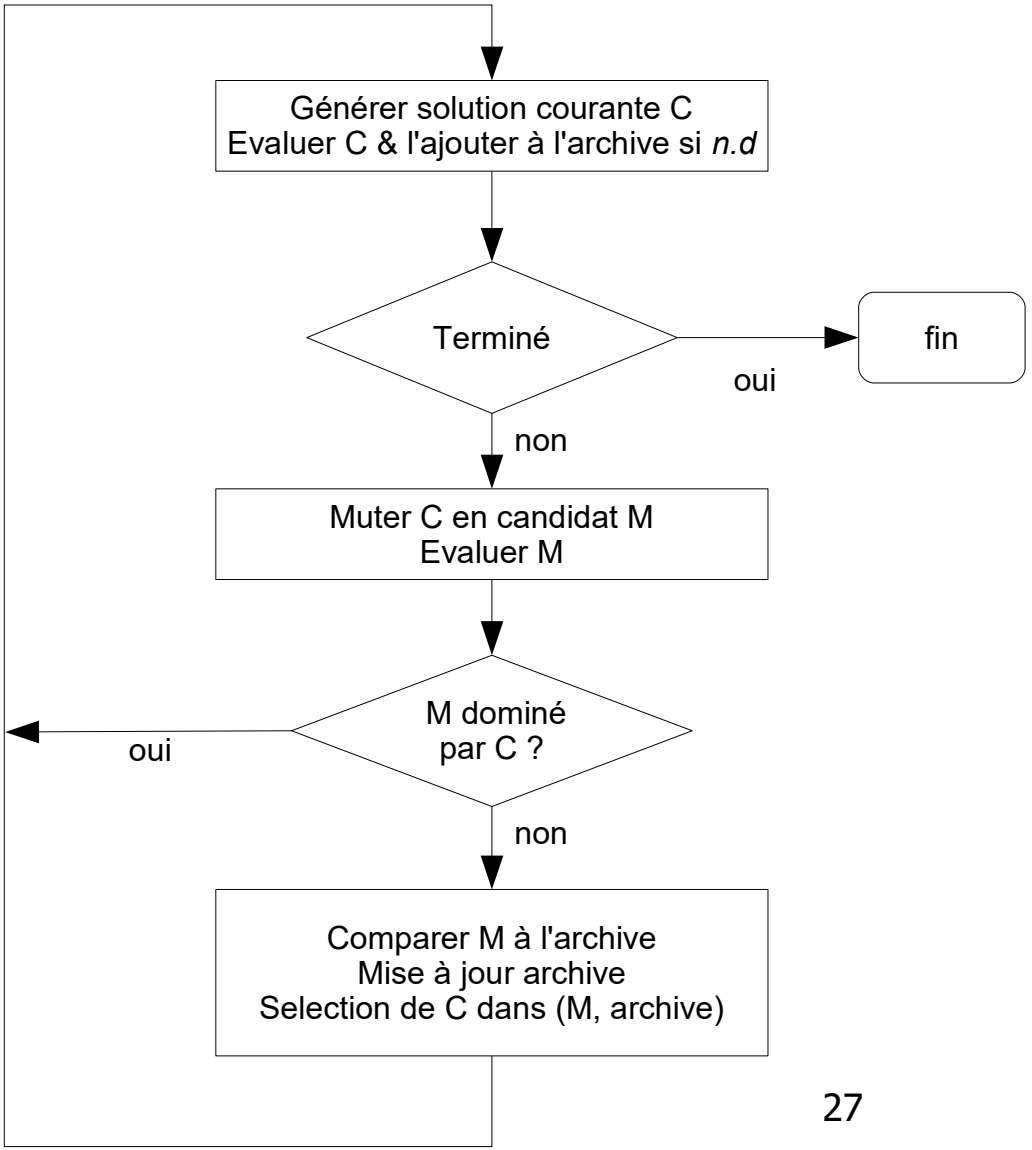
Et aussi en non-Pareto ...
Méthode **lexicographique**
Trier les objectifs par ordre d'importance et les minimiser en séquence.

Si le premier n'a que une solution à valeur optimale, le processus s'arrête

MOEA à dominance 1 → 1
Pareto Archived Evolution Strategy

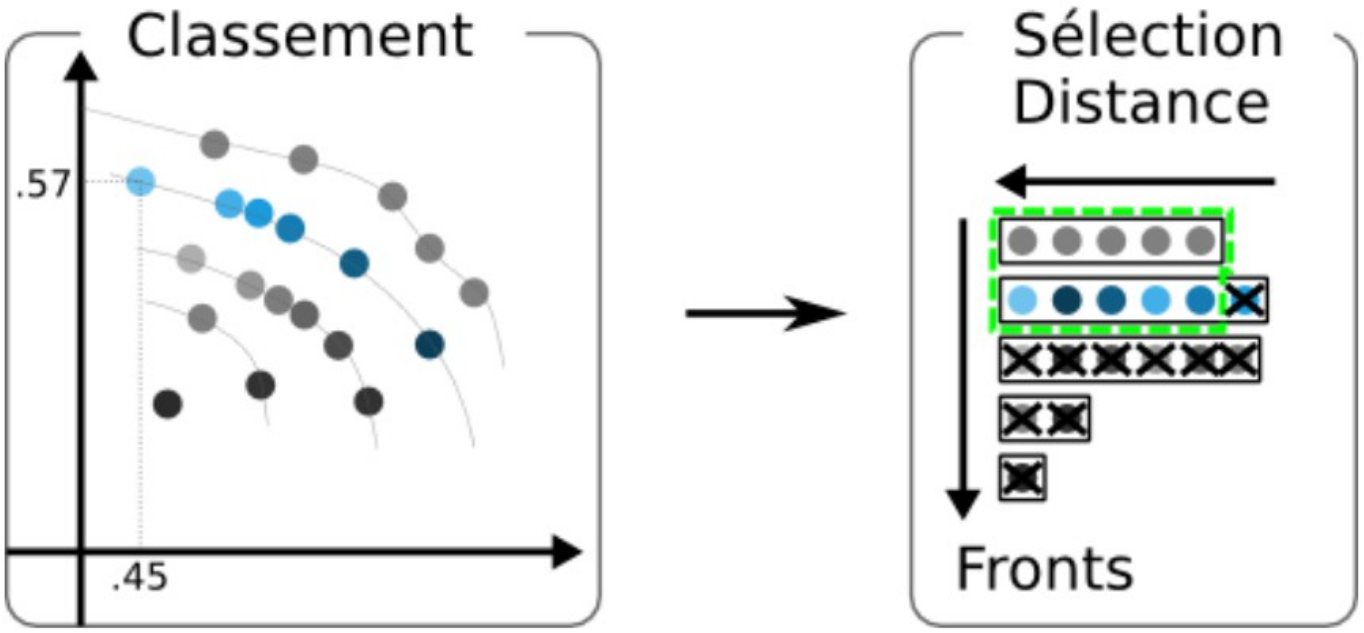
- Evolution d'un individu par mutation
  - Archive de taille fixe pour stocker les non-dominés les nouveaux individus sont confrontés à ceux dans l'archive
  - Division de l'espace des solutions en grille :

    si archive pleine alors élimination d'un individu en zone dense



MOEA à dominance avec population
NSGA-II

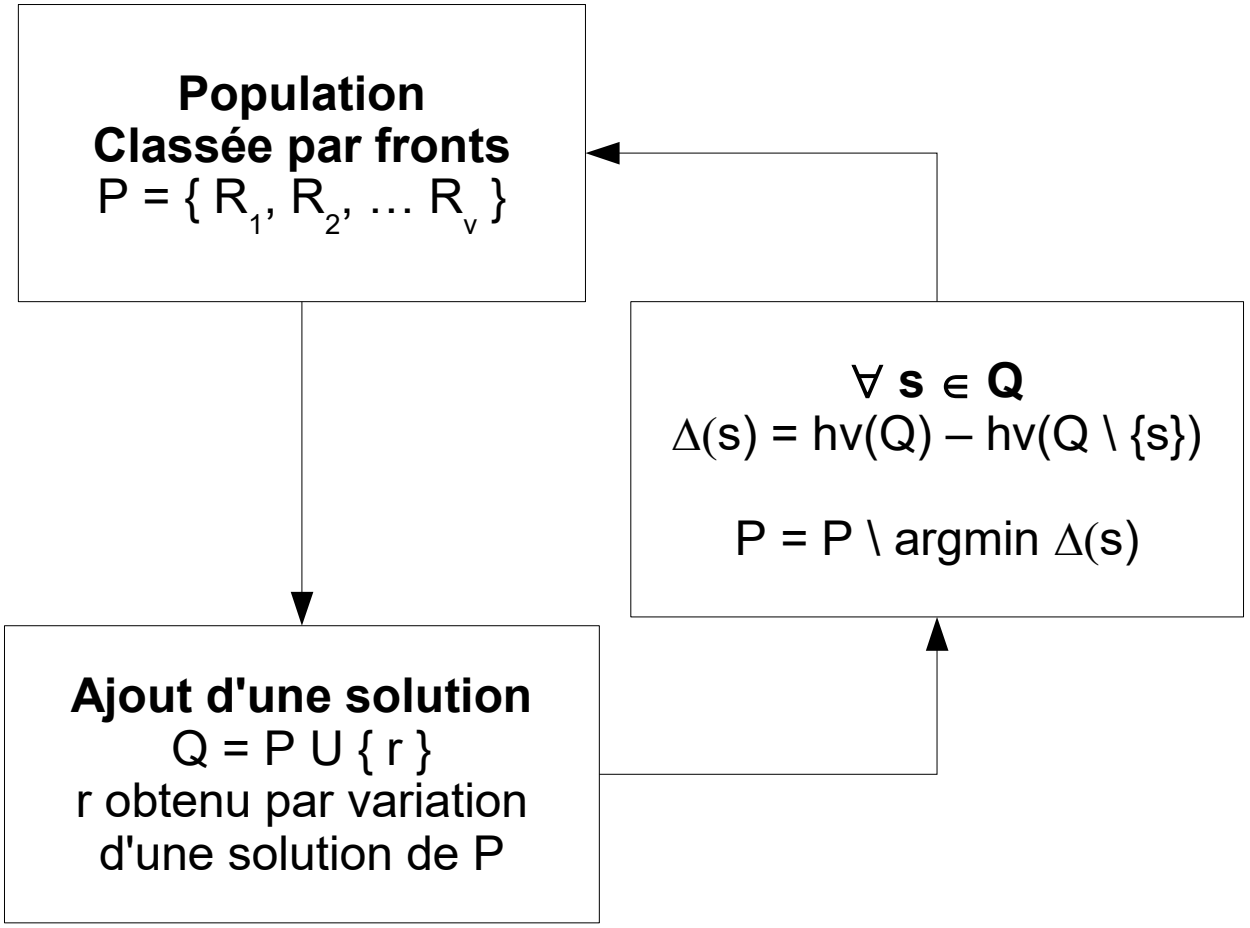
- Reproduction élitiste
  - Les fronts dominants d'abord
  - Les individus les plus isolés sur chaque front

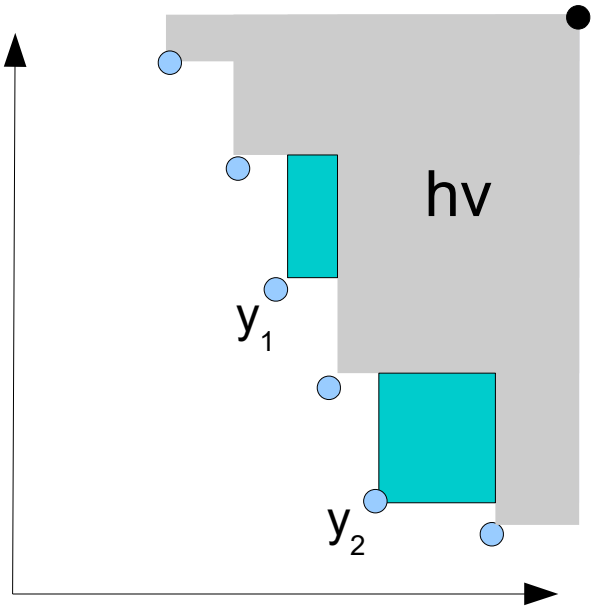


http://www.pinville.com/algorithmes-evolutionnistes.php

MOEA sur indicateur avec population
SMS-EMOA

- La qualité de l'hypervolume produit si on accepte une solution guide la recherche
  - On enlève s_1 ou s_2 ?

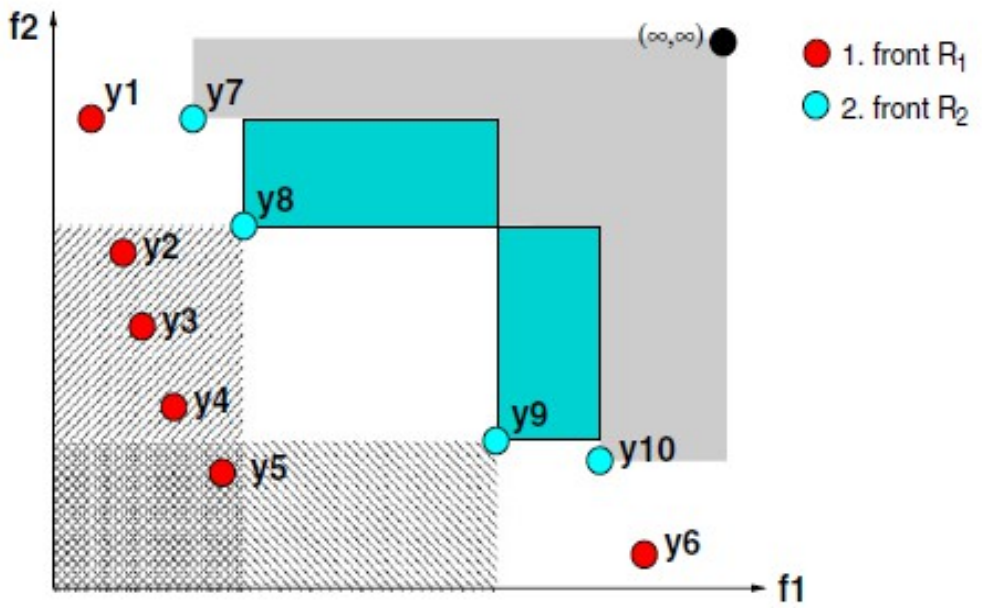




MOEA sur indicateur avec population
SMS-EMOA

- La qualité de l'hypervolume produit si on accepte une solution guide la recherche
  - Classement par front comme avec NSGA-2
  - Si plusieurs fronts, on enlève un point du dernier front
  - Calcul d'hv coûteux → retrait basé sur le nombre de points dominants

s_9 contribue moins que s_8
au hv, mais est dominé par moins de points
→ retrait de s_8

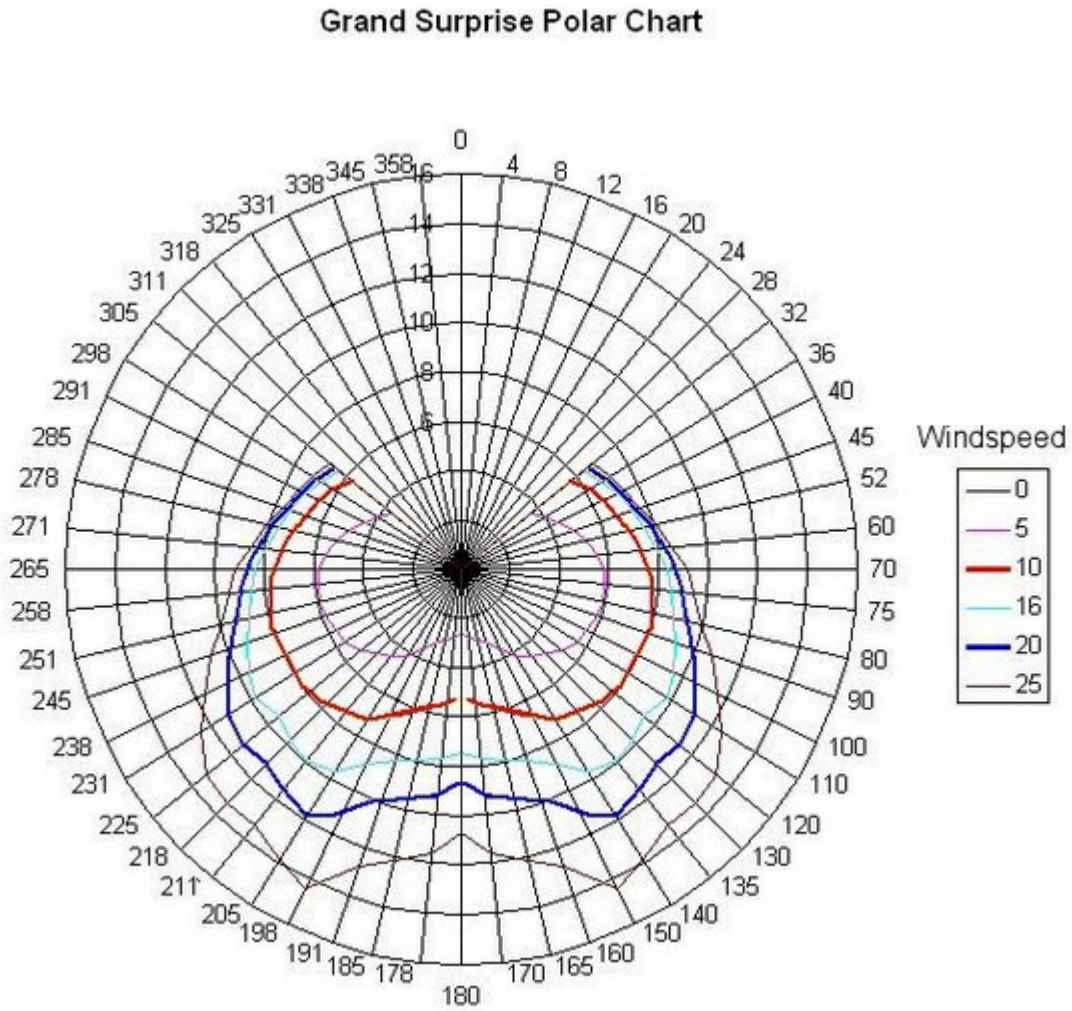


Applications
Routage de bateau

- Trouver la meilleure route pour un bateau
  - Vitesse du bateau en fonction de
    - L'angle au vent
    - La force du vent

  - La météo :

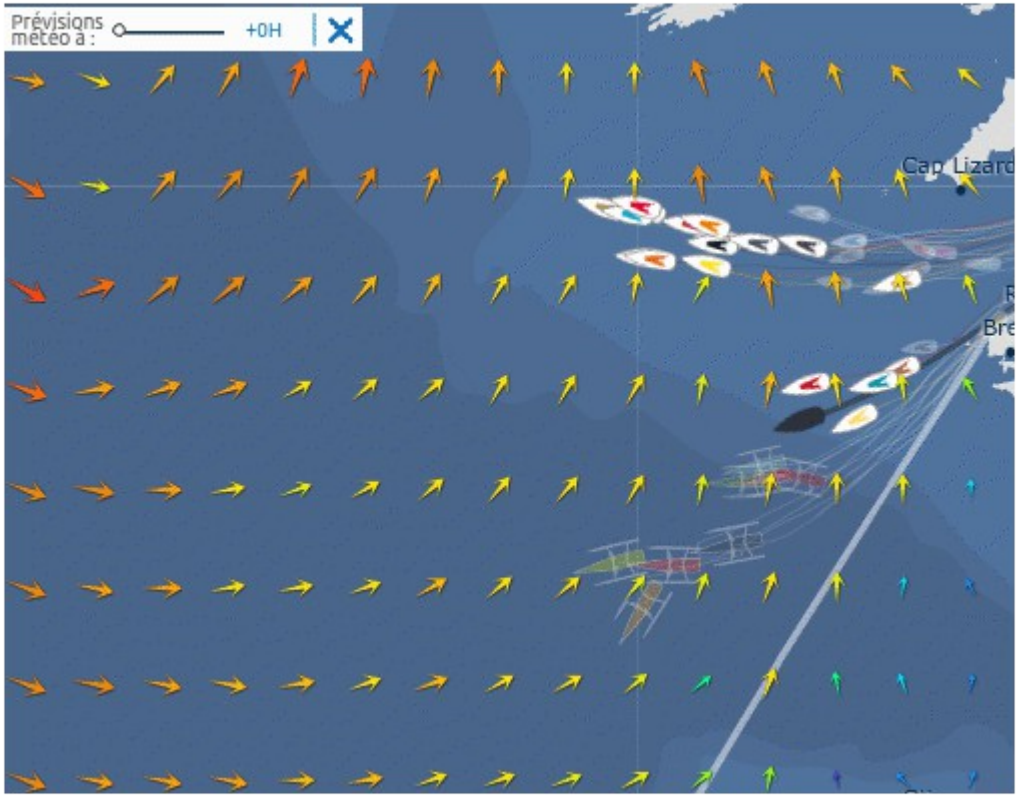
    force et durection du vent au temps t



Applications
Routage de bateau

- Trouver la meilleure route pour un bateau de course à la voile
  - Vitesse du bateau en fonction de
    - L'angle au vent
    - La force du vent
  - La météo :

    force et durection du vent au temps t



Routage
Calcul d'isochrones

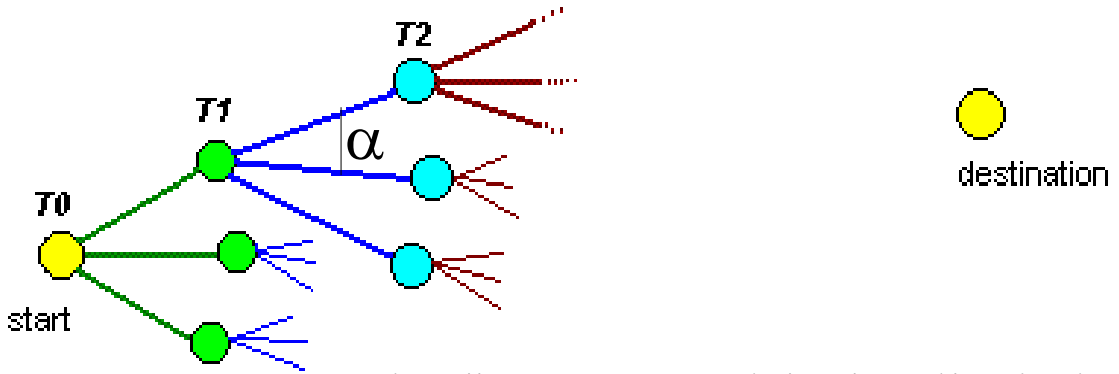
- Principe
  - A partir d'un point (x,y, t), calculer tous les points atteignables au temps t + Δt
  - Suivant chaque direction possible (pas d'angle $\Delta\alpha$)

$\forall$ boatDir = k.$\Delta\alpha$

(windDir, winSpeed) = weather(x,y,t)

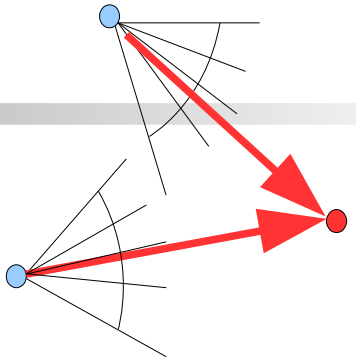
boatSpeed = polar(windSpeed, windDir, boatDir)

(x',y', t'=t + Δt) = addVector(xy, boatDir, boatSpeed*Δt)

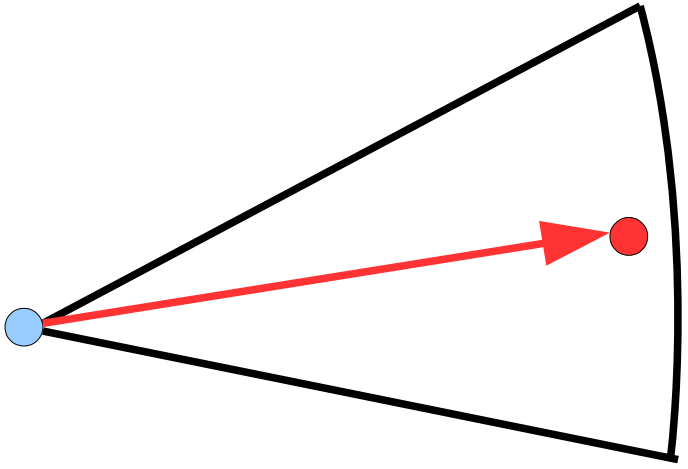


http://www.tecepe.com.br/nav/vrtool/routing.htm

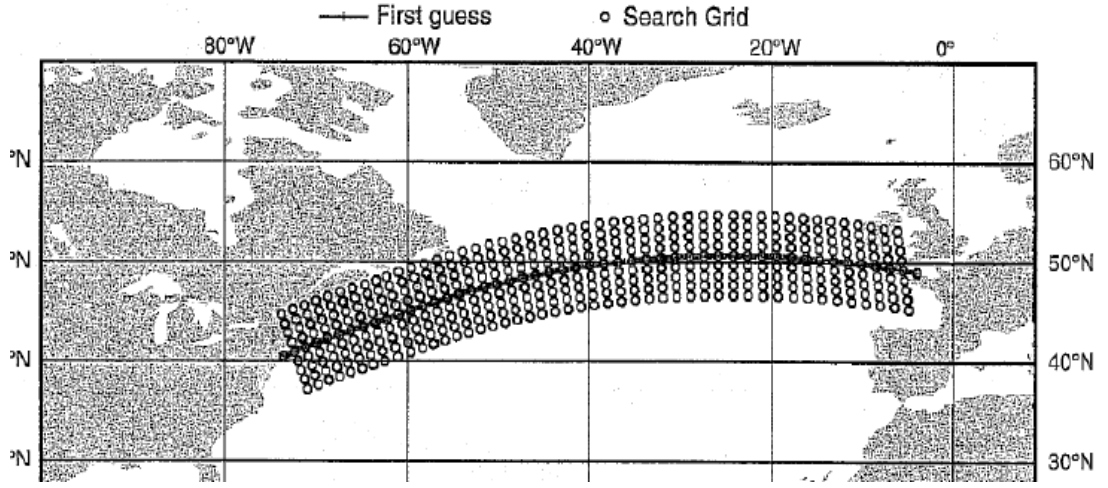
Routage
Calcul d'isochrones



- Coupes dans l'arbre de recherche
  - Angles admissibles à partir de chaque point
  - Zone géographique admissible
  - Sillage



  - Découpage en grille (ε-voisinage)
    - Prog. dynamique

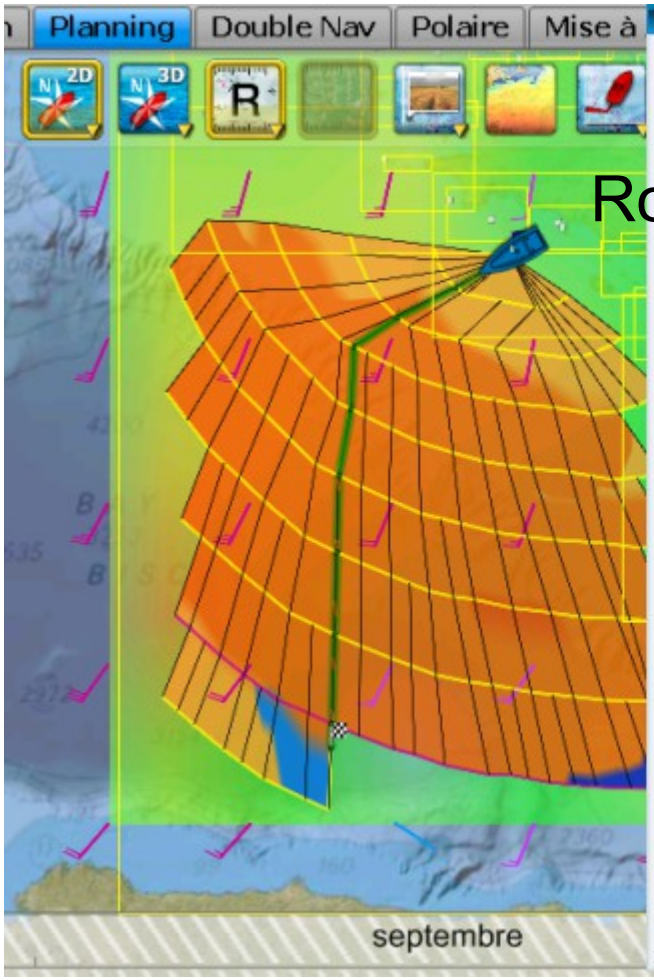


Routage exemple

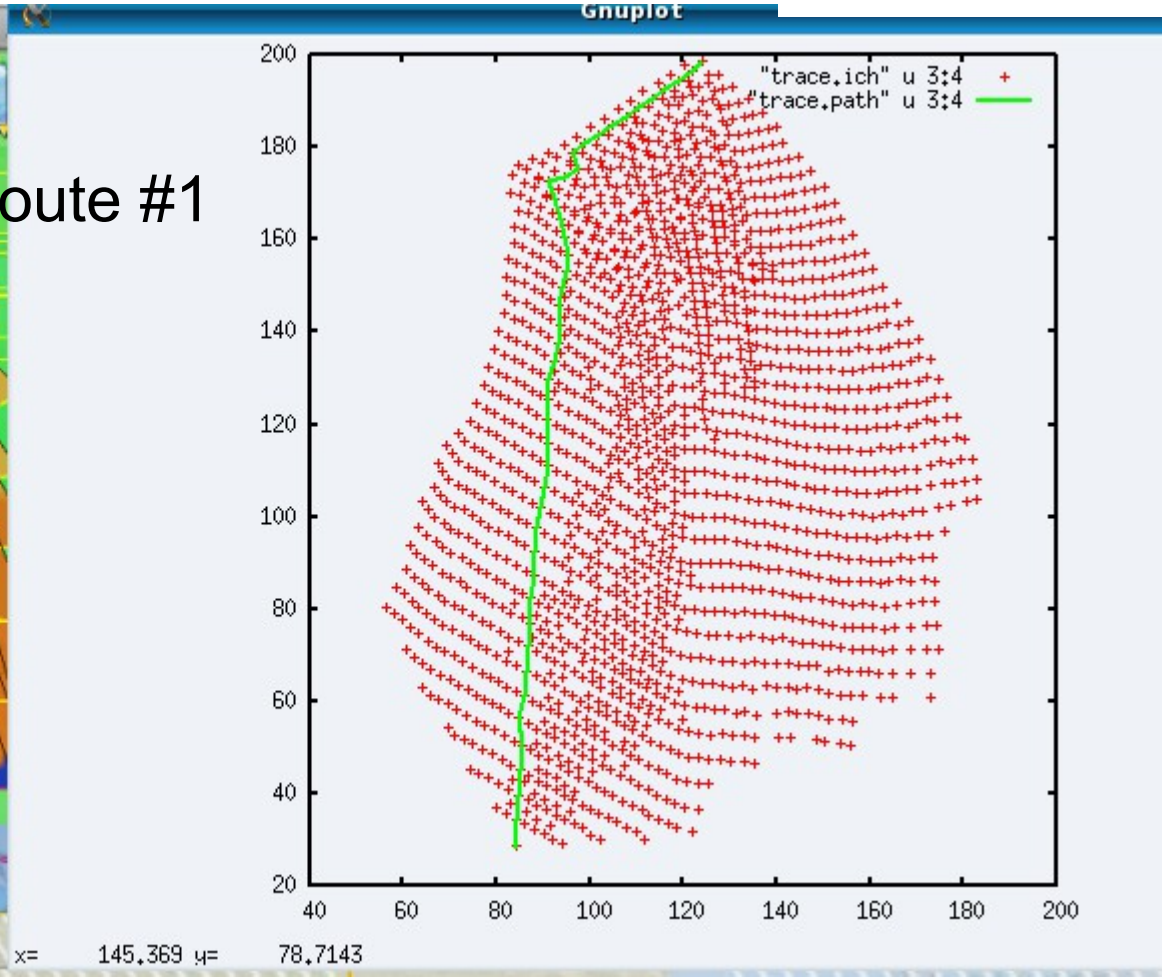
TABLE I
RESULTS FOR 6 ROUTES

route	time to destination	
	MaxSea	IsoLolo
#1	1j03h00	0j21h44
#2	0j20h52	0j18h36
#3	1j14h40	1j13h10
#4	1j01h52	1j00h43
#5	1j06h45	1j07h22
#6	0j18h53	0j18h28

- Comparaison avec Maxsea
  - Δt = 3h, polaire Grand Surprise, temps 6h



Route #1

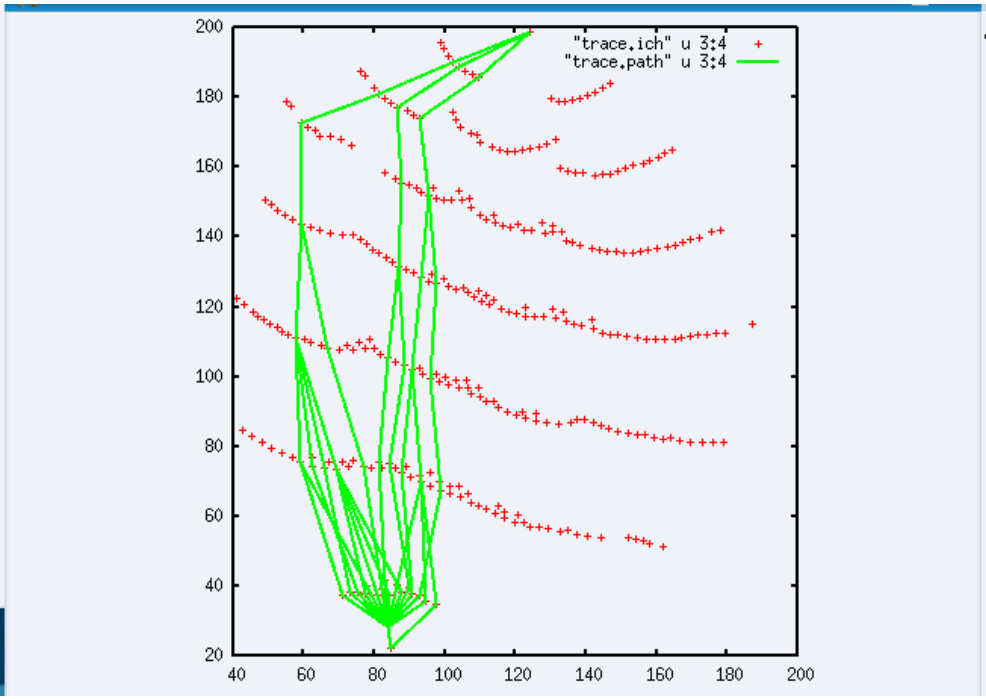


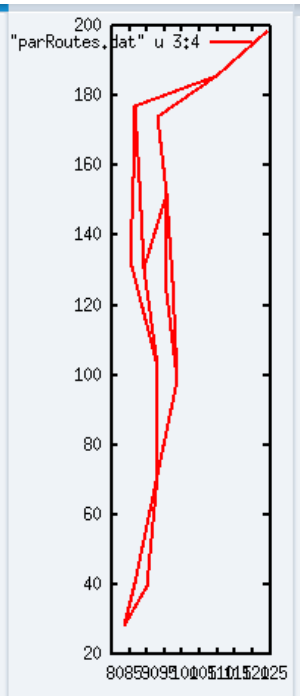
Routage
Front de Pareto

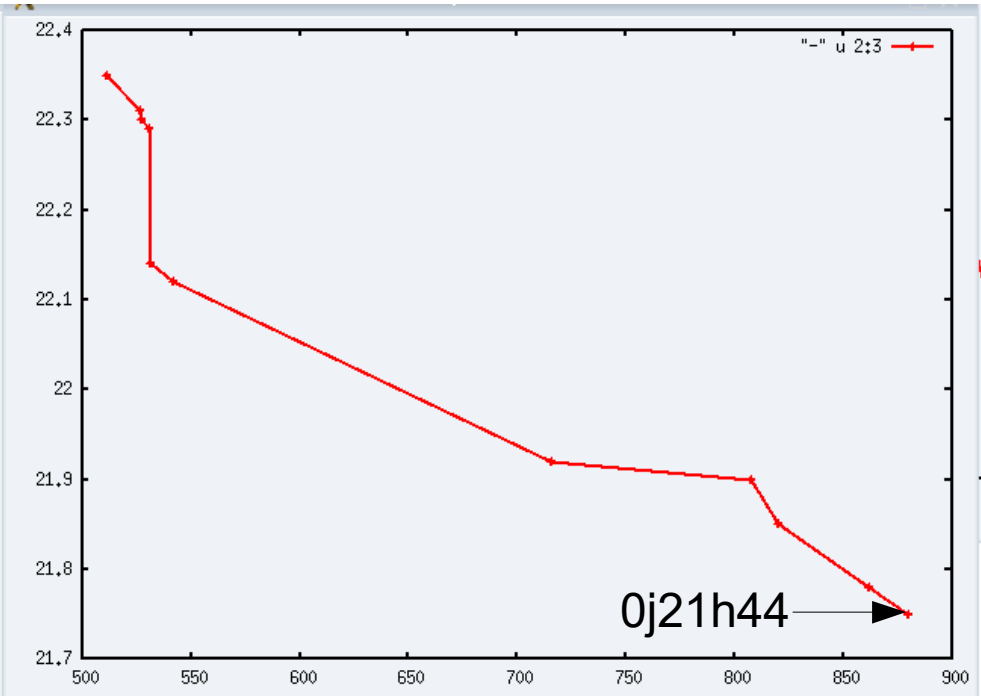
TABLE I
RESULTS FOR 6 ROUTES

route	time to destination	
	MaxSea	IsoLolo
#1	1j03h00	0j21h44
#2	0j20h52	0j18h36
#3	1j14h40	1j13h10
#4	1j01h52	1j00h43
#5	1j06h45	1j07h22
#6	0j18h53	0j18h28

- Compromis temps de parcours ↔ distance parcourue
  - Δt = 3h, polaire Grand Surprise, temps 6h
  - (1) Isochrone → (2) graphe de routes compatibles → (3) front de Pareto







Route #1 (1) (2) (3)

Applications
Groupement de fonctions en taches OS temps réel

- Ordonnancement temps-réel
  - Ensemble de fonctions à exécuter U $F_i = (C_i, T_i, D_i)$
    - C_i : temps max d'execution (WCET)
    - T_i : période d'activation
    - D_i : deadline
  - Implantées en tâches U A_i

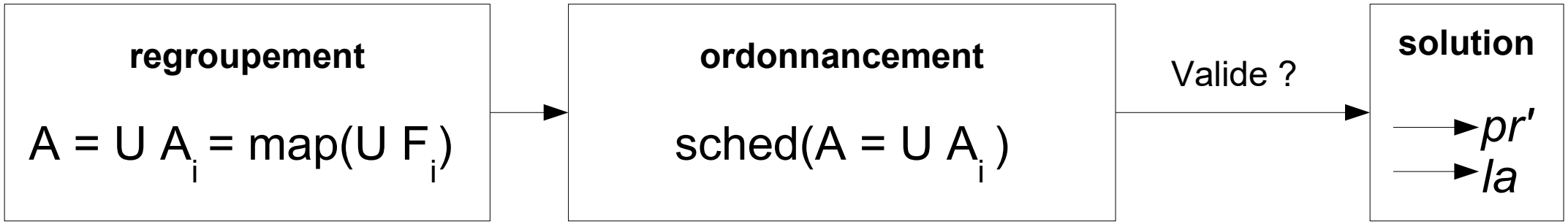
Règles de composition

si $A_k = \{F_i, F_j\}$

$(C_k, T_k, D_k) = (C_i, T_i, D_i) o (C_j, T_j, D_j)$

Applications
Groupement de fonctions en taches OS temps réel

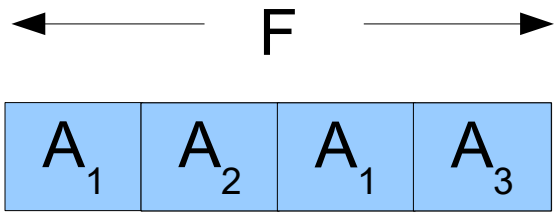
- Optimisation lors de l'ordonnancement
- Préemption *pr* : interruption d'une tâche pour en exécuter une autre, plus prioritaire
  → à minimiser
- Laxité *la* : temps entre la fin de l'exécution d'une tâche et sa deadline (D_i - C_i)
  → à maximiser (minimisation *pr'* = *borneMax* – *pr*)



http://www.pinville.com/algorithmes-evolutionnistes.php

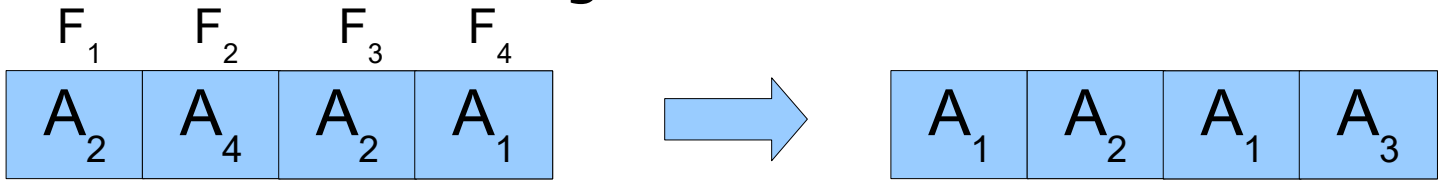
Groupement de fonctions en taches OS temps réel
codage

- Avec PAES : représentation par chromosome des sols.
  - Codage



sol[i] = j
si fonction F_i dans tâche A_j

  - Uniformisation du codage : éviter les redondances



F_1 toujours affectée à T_1

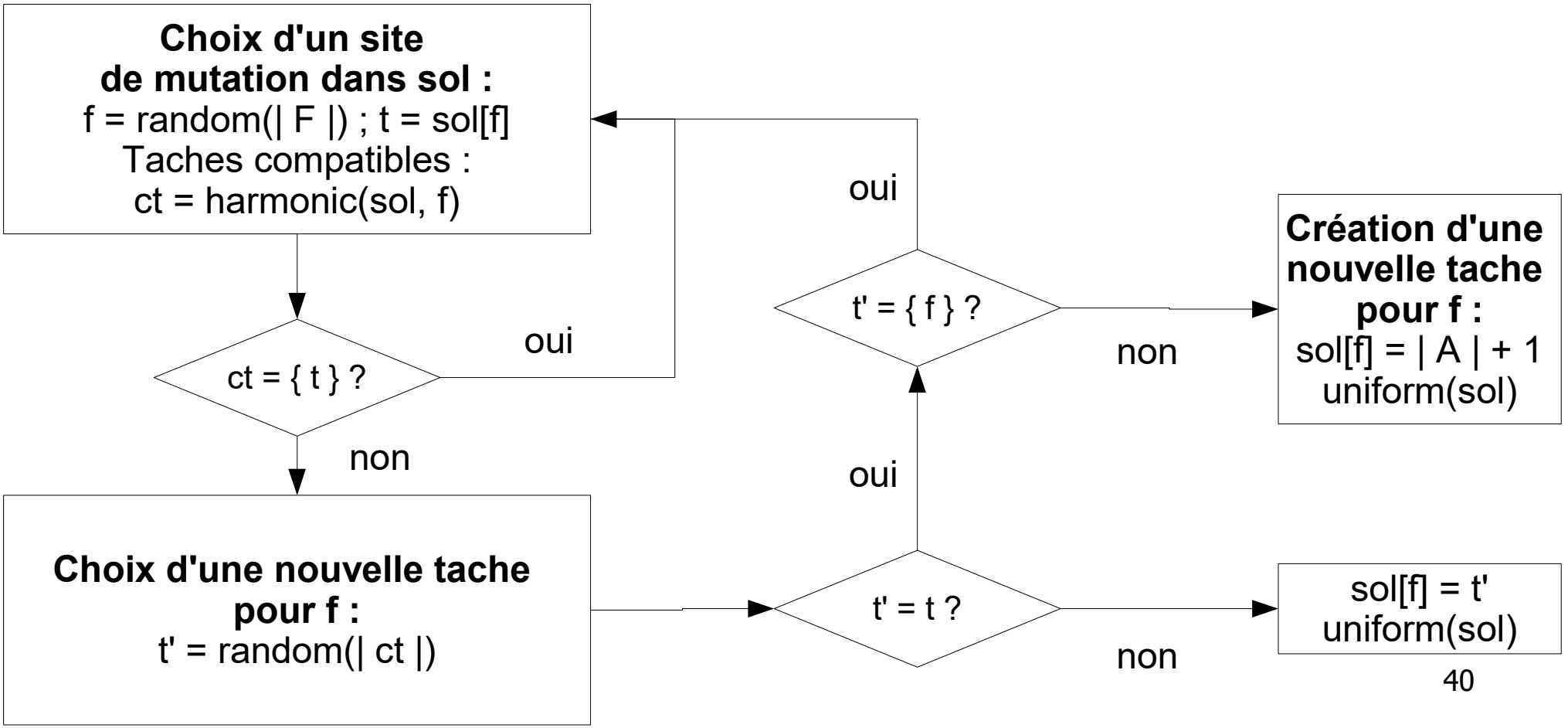
F_2 à T_2, sauf si groupée avec F_1 dans T_1

F_3 à T_3 sauf si dans T_1 ou T_2

...

Groupement de fonctions en taches OS temps réel
mutation

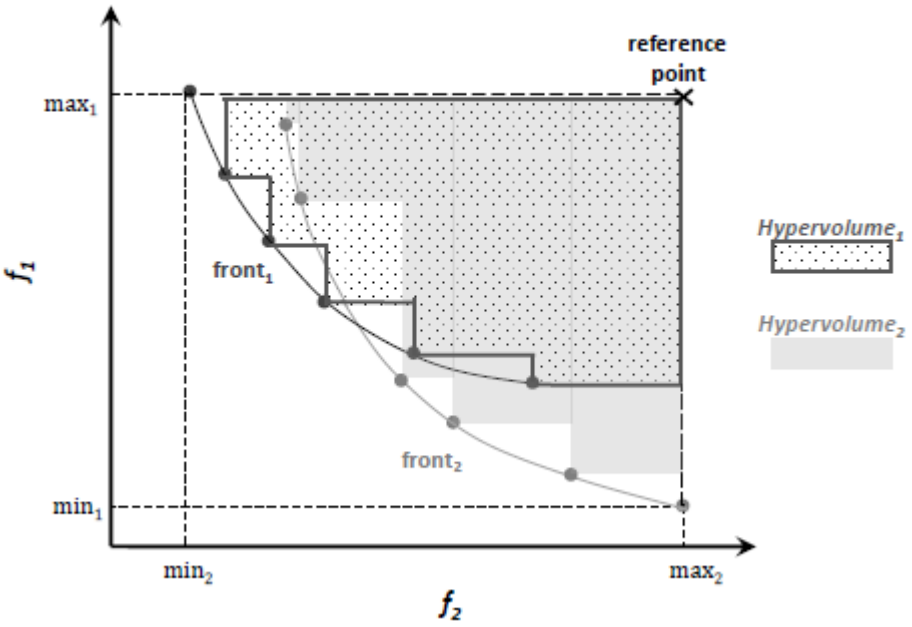
- Doit respecter la compatibilité entre taches/fonctions, qui doivent être harmoniques



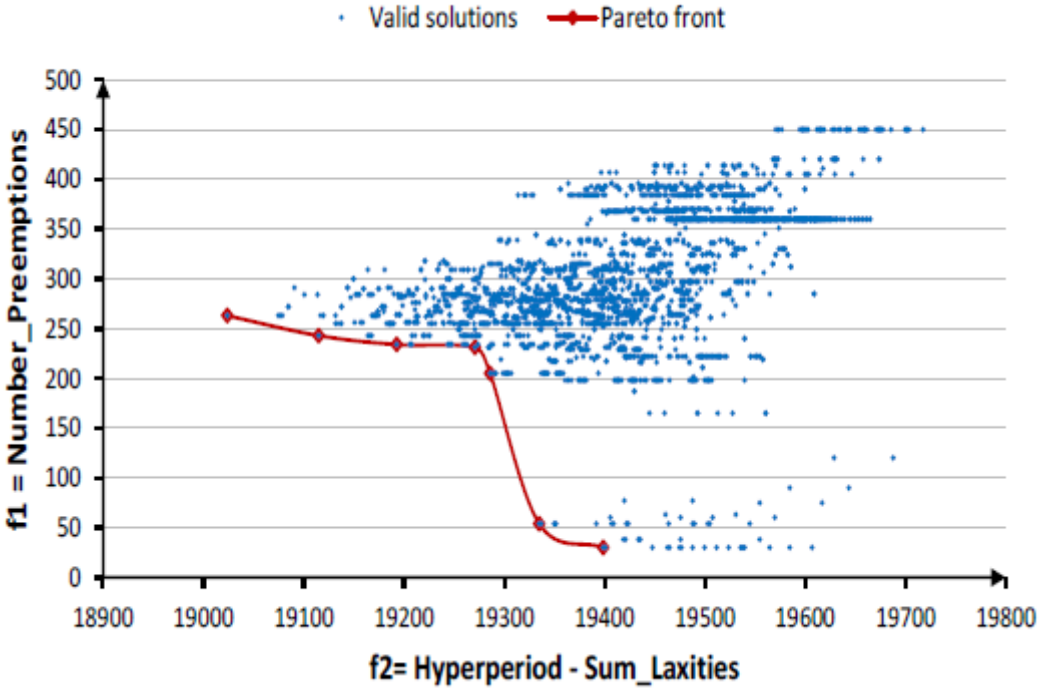
Groupement de fonctions en taches OS temps réel
Comparaisons: hv et normalisation des objectifs

- Pour que un objectif ne prenne pas le pas sur l'autre lors d'un calcul d'hypervolume. Ex : ici une petite variation de f1 améliore bien plus l'hypervolume que la même pour f2.

$$x' = \frac{x - min_1}{max_1 - min_1} \qquad y' = \frac{y - min_2}{max_2 - min_2}$$

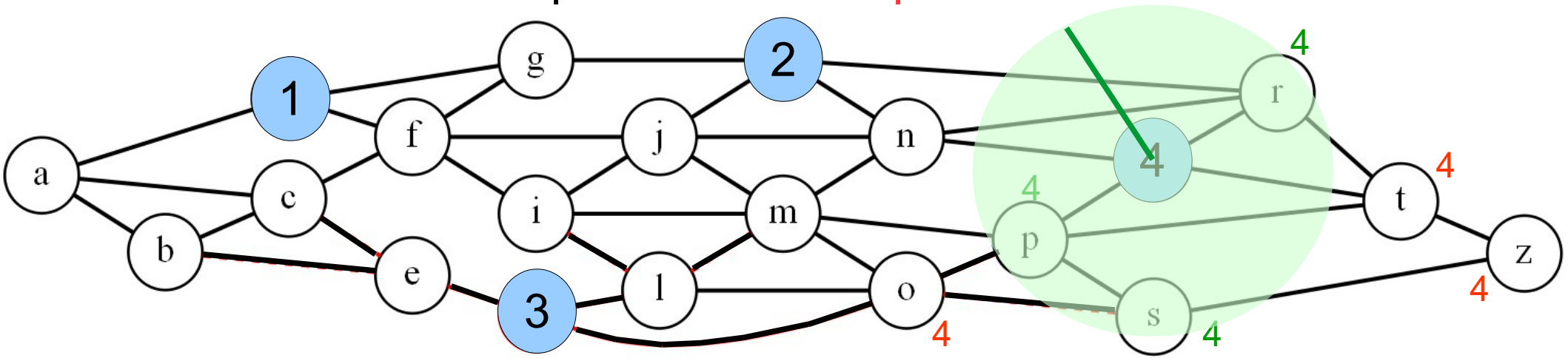


Toutes les solutions valides d'un cas de test avec 11 fonctions



Applications
Localisation d'écoles

- Optimisation en fonction de la population (enfants allant à l'école)
  - P écoles à placer, avec N points de population, et les distances entre les différents sites. Si un enfant est à moins de T km de l'école la plus proche, il y va à pied, sinon, il prend les transports en commun. Objectifs :
    - Distance moyenne pour ceux qui vont à pied
    - Nombre de ceux qui vont en transport



Localisation d'écoles

résolution exacte ε-contraintes

- Définition du problème
  - $Y_{ij} = 1$
    enfant i à école j
  - $Z_{ij} = 1$
    enfant i à école j **à pied**
- Résolution par itération sur la borne #byPublic transportation

$$\text{minimize} \sum_{i=1}^{D}\sum_{j=1}^{N} h_i d_{ij} Z_{ij}$$ ← o_1 : distance totale à pied

$$\text{minimize} \sum_{i=1}^{D}\sum_{j=1}^{N} h_i (Y_{ij} - Z_{ij})$$ ← o_2 : enfants en transports publics

$$\text{subject to} \sum_{j=1}^{N} Y_{ij} = 1 \qquad \forall i, 1 \le i \le D$$

$$\sum_{j=1}^{N} X_j = p$$

$$Y_{ij} - X_j \le 0 \qquad \forall i, j,\ 1 \le i \le D,\ 1 \le j \le N$$

$$Z_{ij} = \begin{cases} 0 & \text{if } d_{ij} > \alpha \\ Y_{ij} & \text{otherwise} \end{cases} \qquad \forall i, j,\ 1 \le i \le D,\ 1 \le j \le N$$

$$\sum_{\substack{k=1 \\ d_{ik} \le \alpha}}^{N} Y_{ik} \ge X_j \qquad \forall i, j,\ 1 \le i \le D,\ 1 \le j \le N, d_{ij} \le \alpha,$$

$$X_j \in \{0,1\} \qquad \forall j,\ 1 \le j \le N$$

$$Y_{ij} \in \{0,1\} \qquad \forall i, j,\ 1 \le i \le D,\ 1 \le j \le N$$

$$Z_{ij} \in \{0,1\} \qquad \forall i, j,\ 1 \le i \le D,\ 1 \le j \le N$$

Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1m = +\infty$ et $o_2m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1m$
$o_2(P) \geq o_2m$

résoudre (z_2 = min $o_2(P)$, P)

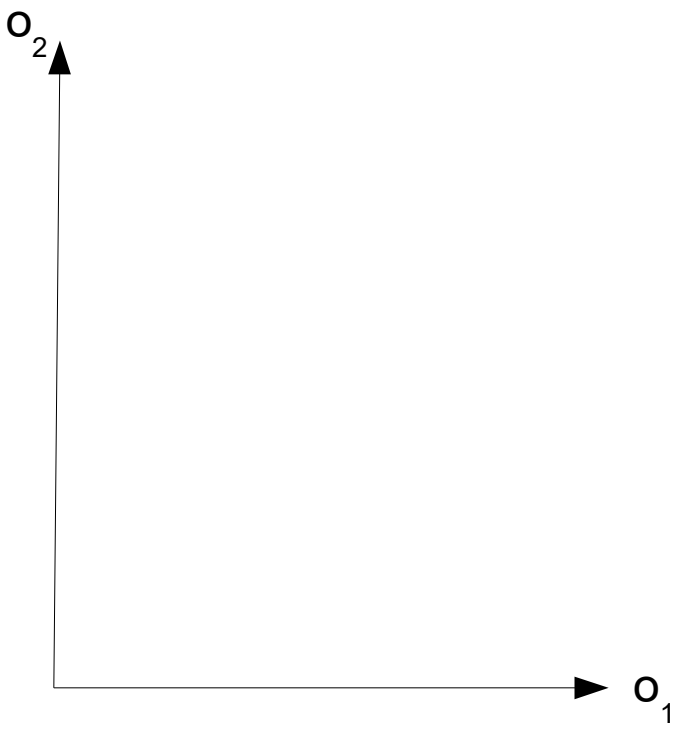
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre (z_1 = min $o_1(P)$, P)

Ajouter (z_1, z_2) à S
$o_1m = z_1 - 1$
$o_2m = z_2 - 1$



Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1 m = +\infty$ et $o_2 m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1 m$
$o_2(P) \geq o_2 m$

résoudre (z_2 = min $o_2(P)$, P)

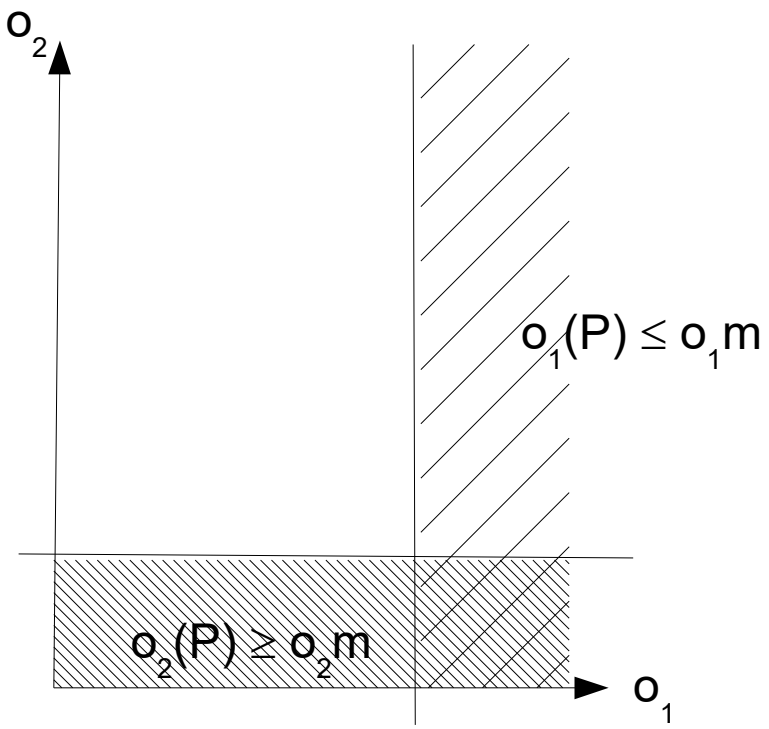
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre (z_1 = min $o_1(P)$, P)

Ajouter (z_1, z_2) à S
$o_1 m = z_1 - 1$
$o_2 m = z_2 - 1$



Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1m = +\infty$ et $o_2m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1m$
$o_2(P) \geq o_2m$

résoudre (z_2 = min $o_2(P)$, P)

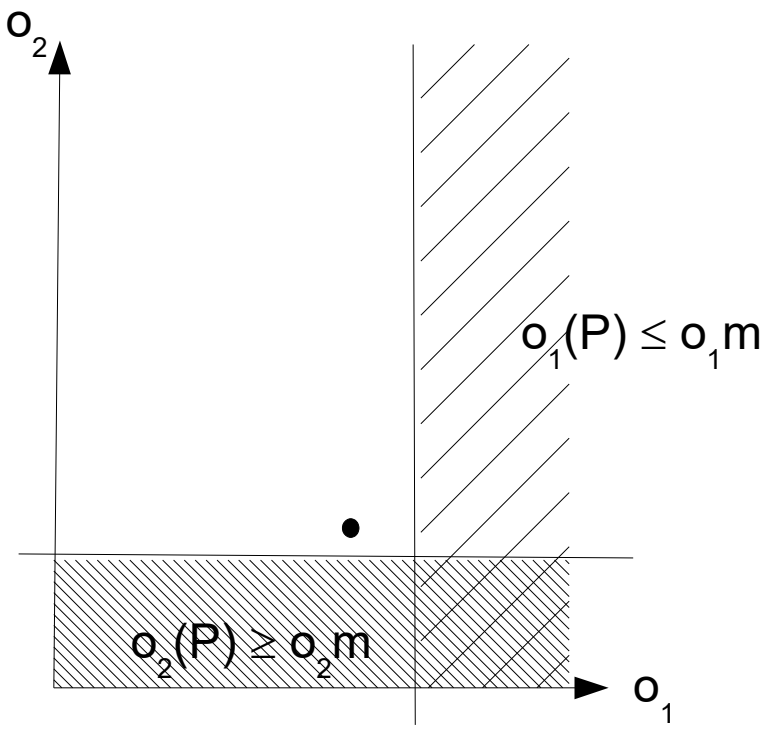
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre (z_1 = min $o_1(P)$, P)

Ajouter (z_1, z_2) à S
$o_1m = z_1 - 1$
$o_2m = z_2 - 1$



Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1 m = +\infty$ et $o_2 m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1 m$
$o_2(P) \geq o_2 m$

résoudre (z_2 = min $o_2(P)$, P)

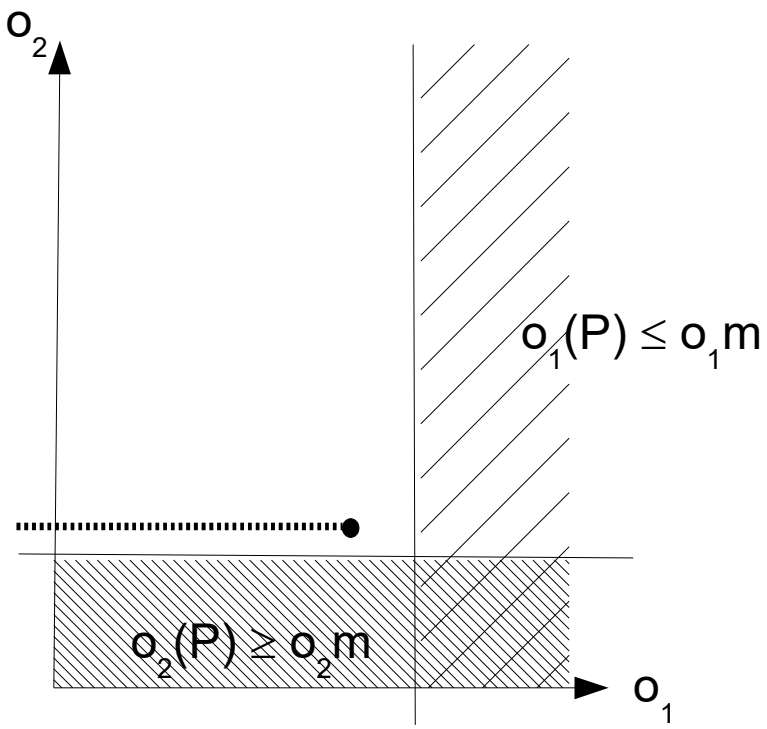
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre (z_1 = min $o_1(P)$, P)

Ajouter (z_1, z_2) à S
$o_1 m = z_1 - 1$
$o_2 m = z_2 + 1$



Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1m = +\infty$ et $o_2m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1m$
$o_2(P) \geq o_2m$

résoudre ($z_2 = \min o_2(P)$, P)

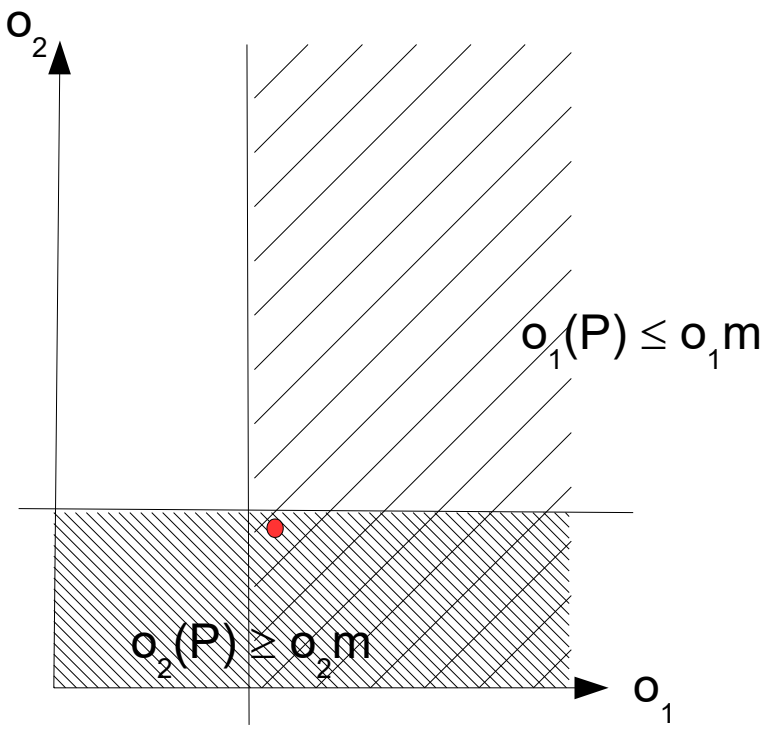
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre ($z_1 = \min o_1(P)$, P)

Ajouter (z_1, z_2) à S
$o_1m = z_1 - 1$
$o_2m = z_2 + 1$



Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1m = +\infty$ et $o_2m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1m$
$o_2(P) \geq o_2m$

résoudre (z_2 = min $o_2(P)$, P)

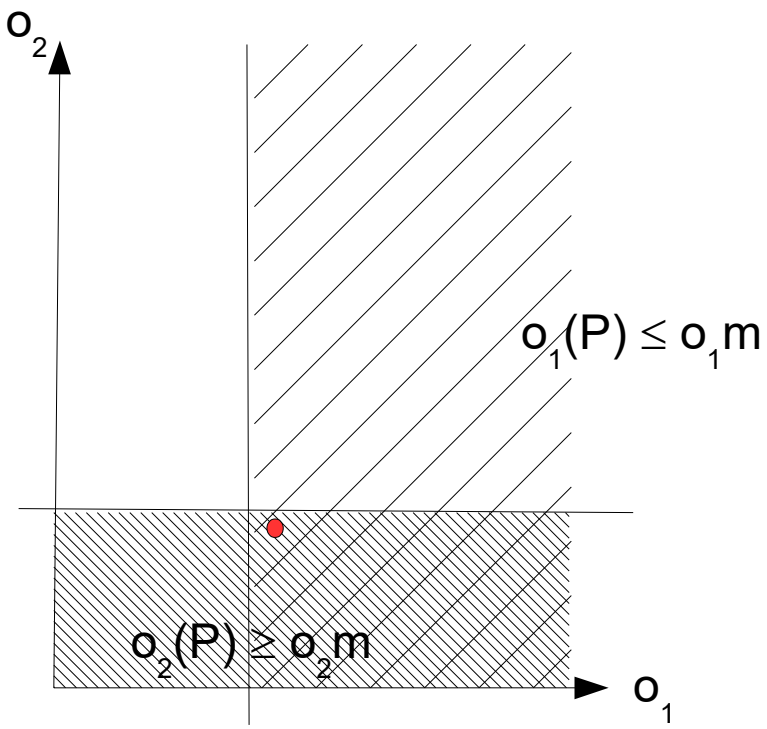
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre (z_1 = min $o_1(P)$, P)

Ajouter (z_1, z_2) à S
$o_1m = z_1 - 1$
$o_2m = z_2 + 1$



Localisation d'écoles

résolution exacte ε-contraintes

initialisation
$o_1m = +\infty$ et $o_2m = -\infty$
P = problème initial
S = $\varnothing$

Ajouter à P les contraintes
$o_1(P) \leq o_1m$
$o_2(P) \geq o_2m$

résoudre ($z_2 = \min o_2(P)$, P)

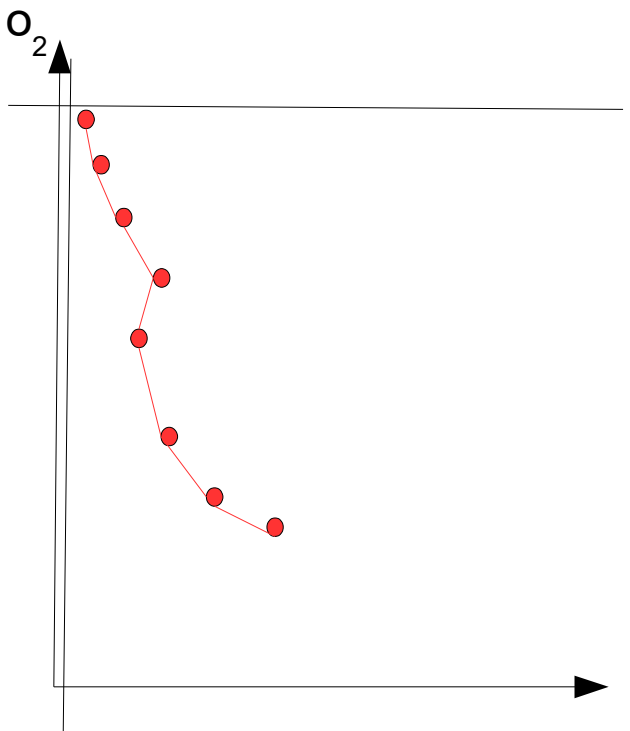
Si P infaisable

oui → fin

non

Ajouter à P la contrainte
$o_2(P) = z_2$
résoudre ($z_1 = \min o_1(P)$, P)

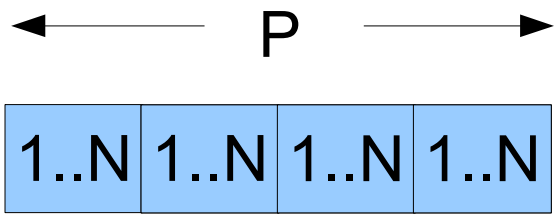
Ajouter (z_1, z_2) à S
$o_1m = z_1 - 1$
$o_2m = z_2 + 1$



Plusieurs heures si N>100 !

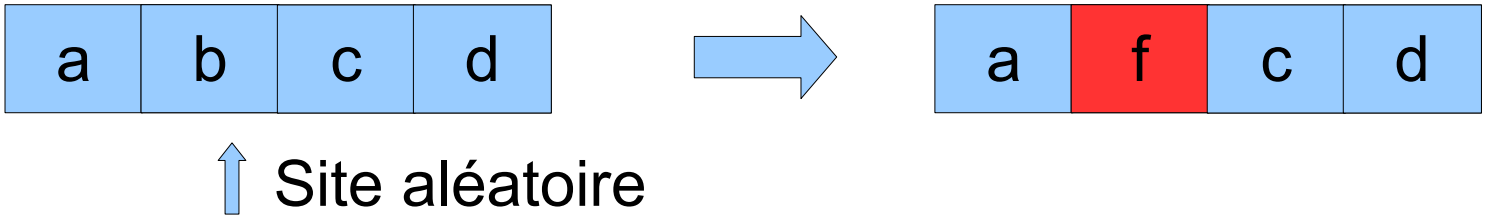
Localisation d'écoles résolution approchée

- Avec PAES ou NSGA2 : représentation par chromosome des sols.
  - Codage

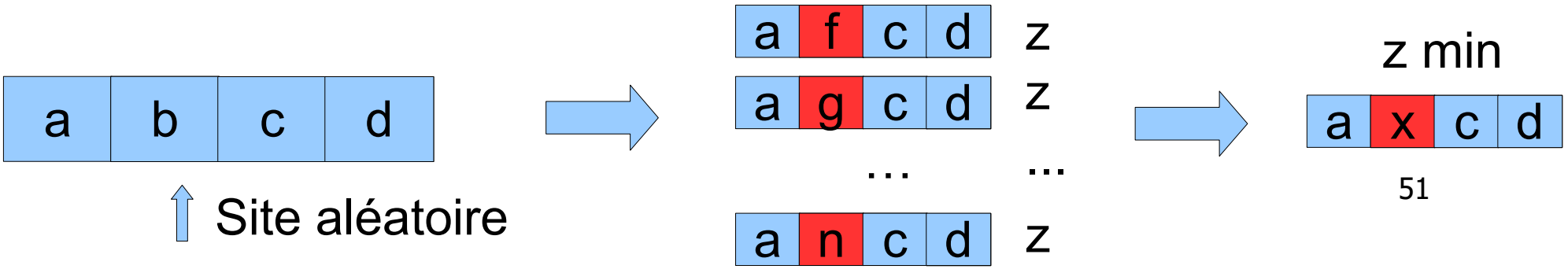


sol[i] = j
si enfant i va à école j

  - Mutation SOO
    1 → 1

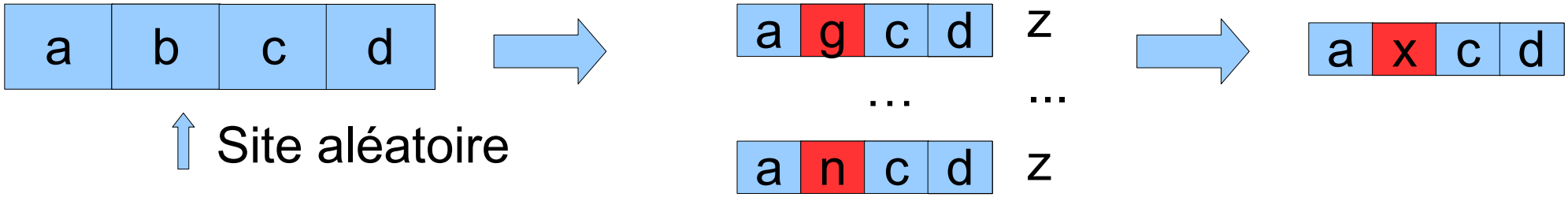


  - 1 → λ (recherche locale : VNS, path relinking)

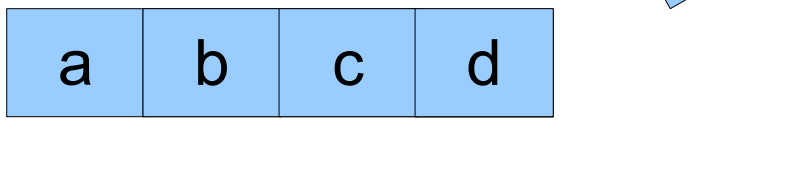


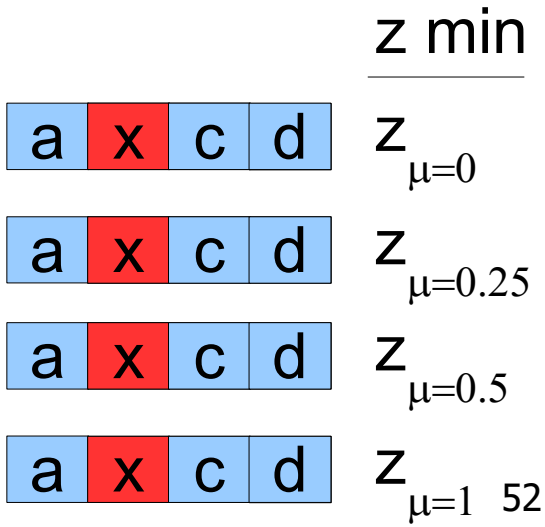
Localisation d'écoles résolution approchée

- Mutation avec recherche locale
  - Du cas SOO ...



  - ... au cas MOO
    - $z_\mu = \mu.z_1 + (1 - \mu).z_2$
    - $\mu = \{0, 0.25, 0.5, 0.75, 1\}$



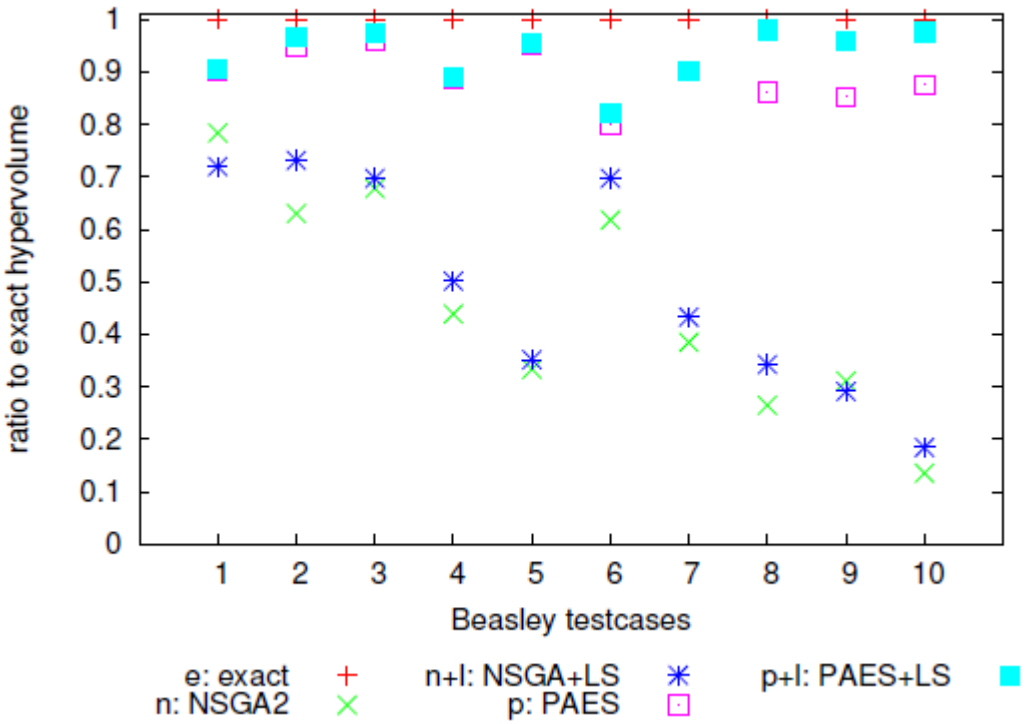


Localisation d'écoles
résolution approchée

- Couplage avec PAES
  - Mutation 1 $\rightarrow \lambda$ à chaque itération
- Couplage avec NSGA
  - Mutation 1 $\rightarrow \lambda$ sur la population finale

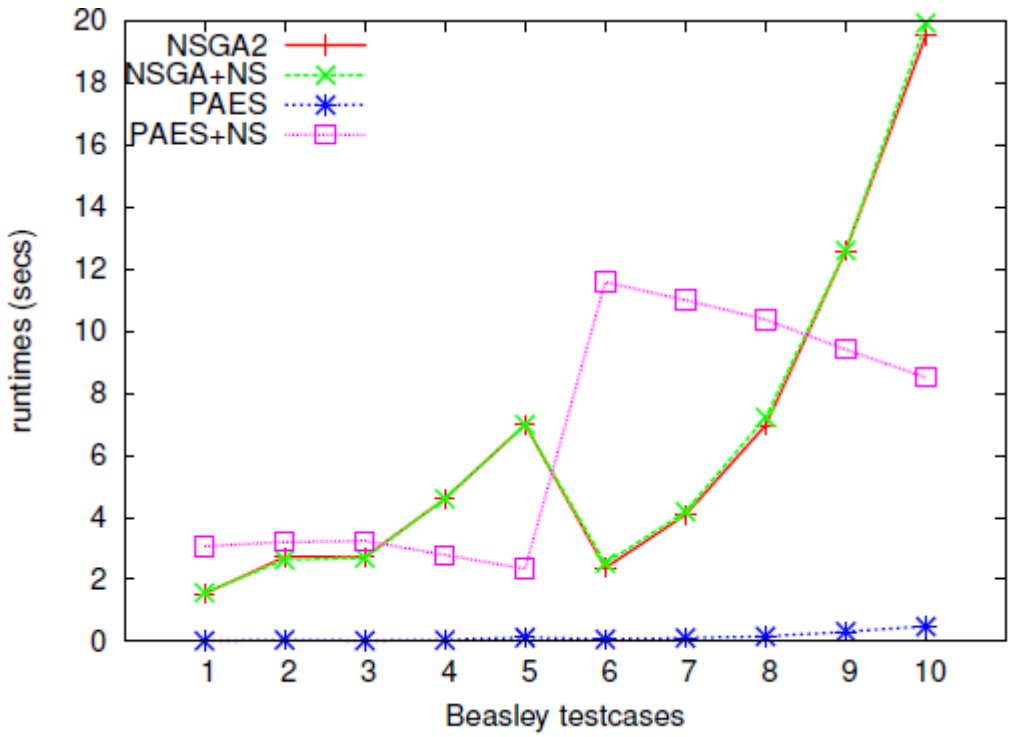
- Comparaison des méthodes
  - Exacte (ε-constraint)
  - PAES (avec/sans LS)
  - NSGA2 (avec/sans LS)

Qualité : ratio $\frac{hv_{méthode}}{hv_{exact}}$

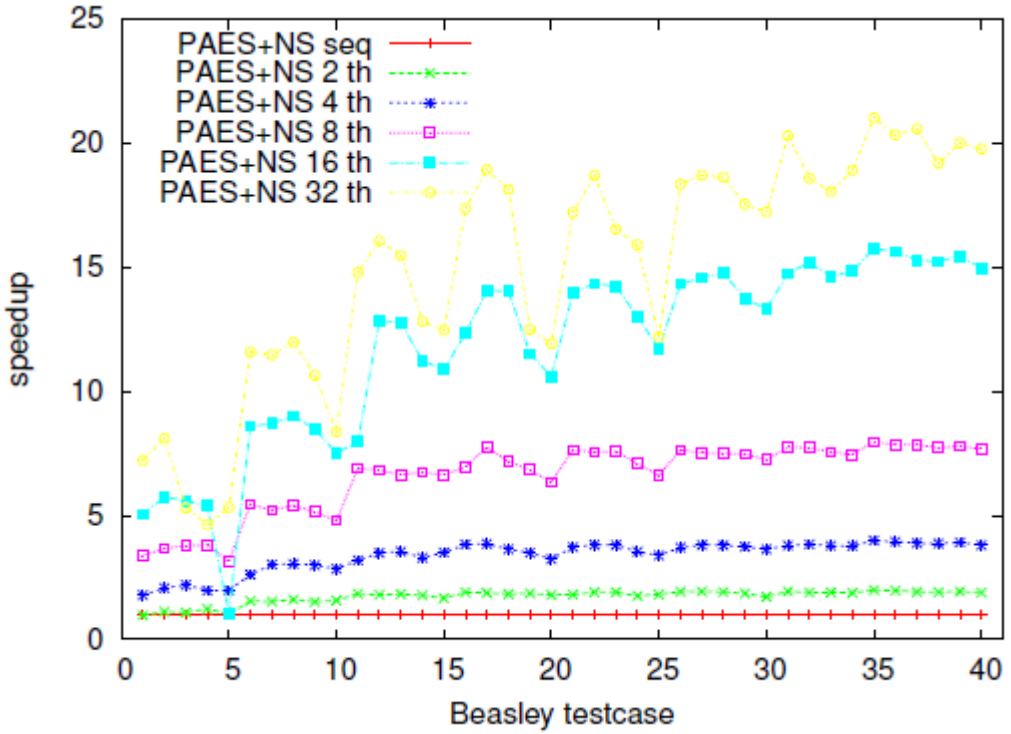


Localisation d'écoles
résolution approchée

Temps d'exécution séquentiel



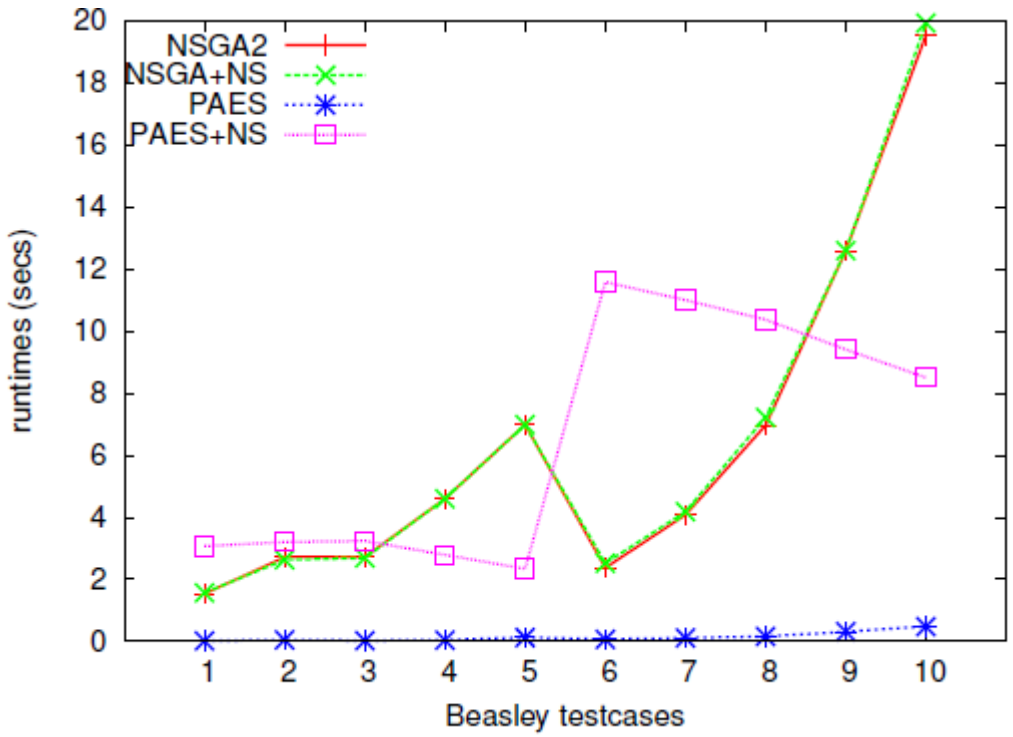
Accélération en parallèle



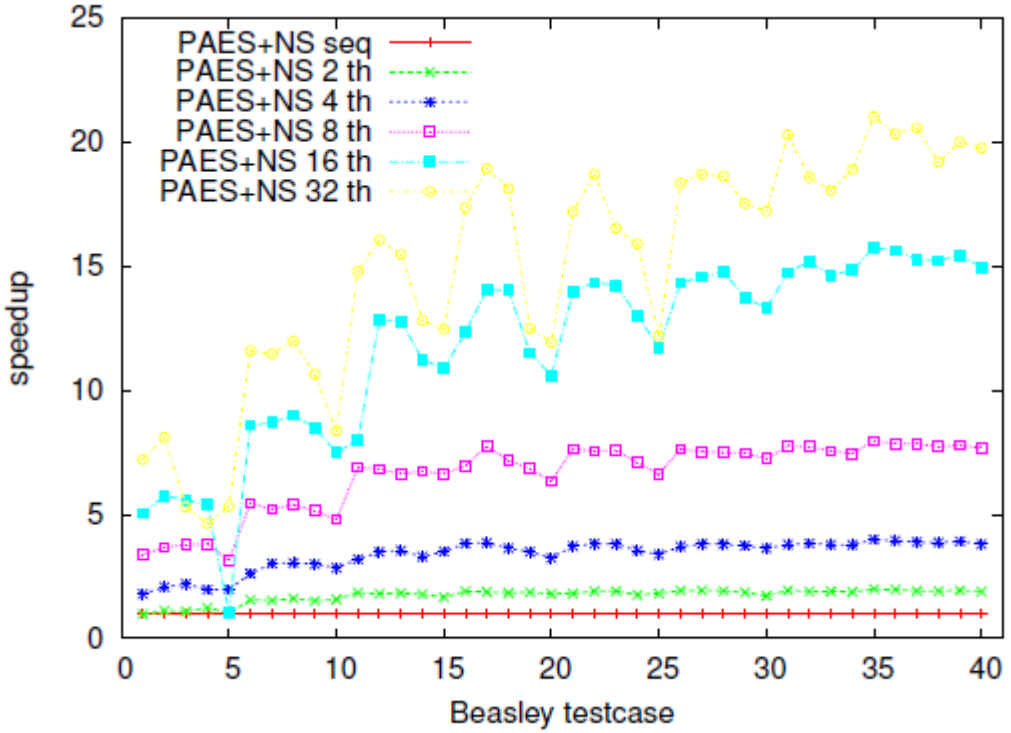
$$\text{Speedup} : \frac{t_{\text{méthode avec x threads}}}{t_{\text{séquentiel}}}$$

Localisation d'écoles
résolution approchée

Temps d'exécution séquentiel



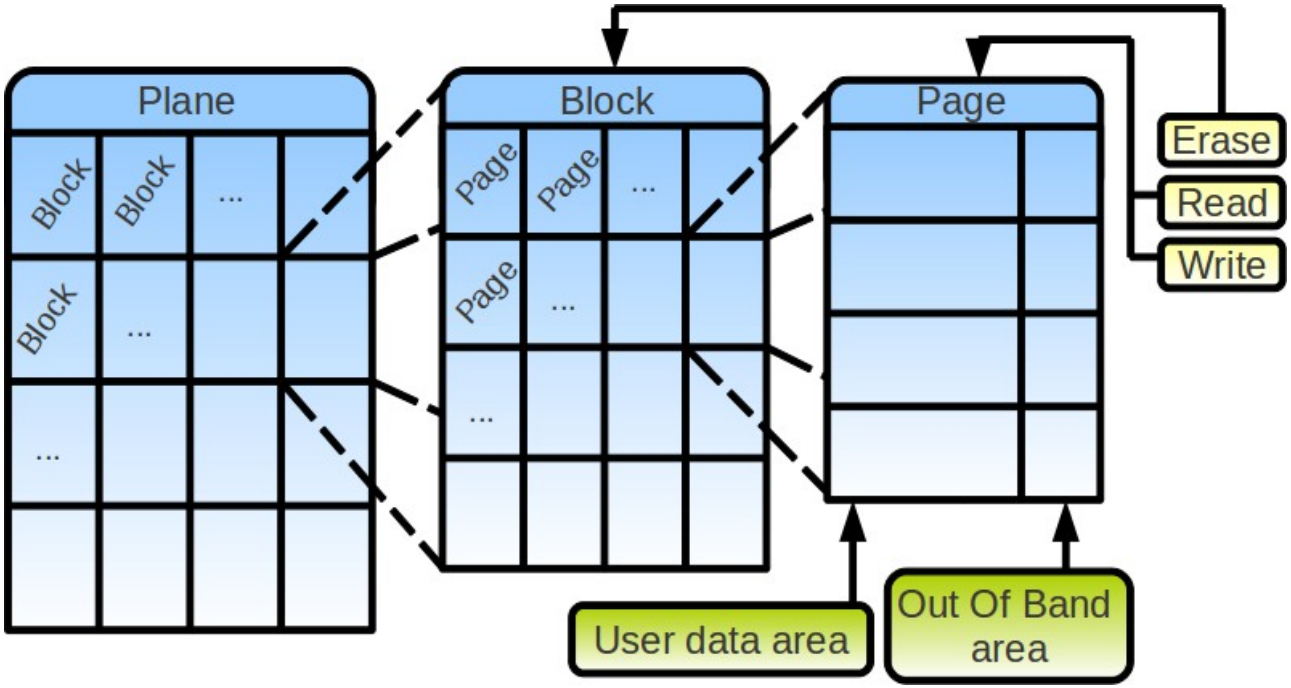
Accélération (24 cœurs)



$$\text{Speedup} : \frac{t_{\text{méthode avec x threads}}}{t_{\text{séquentiel}}}$$

Applications
Configuration de pilote de mémoire Flash

- Actions
  - Opérations E/R/W
  - E sur blocs (usure)
  - R/W sur pages
  - E avant W



mapping des adresses

- Par page (PM) → coût RAM
- Par bloc (BM) → coût #E
- Hybride → %PM

Choix BM vs PM pour W

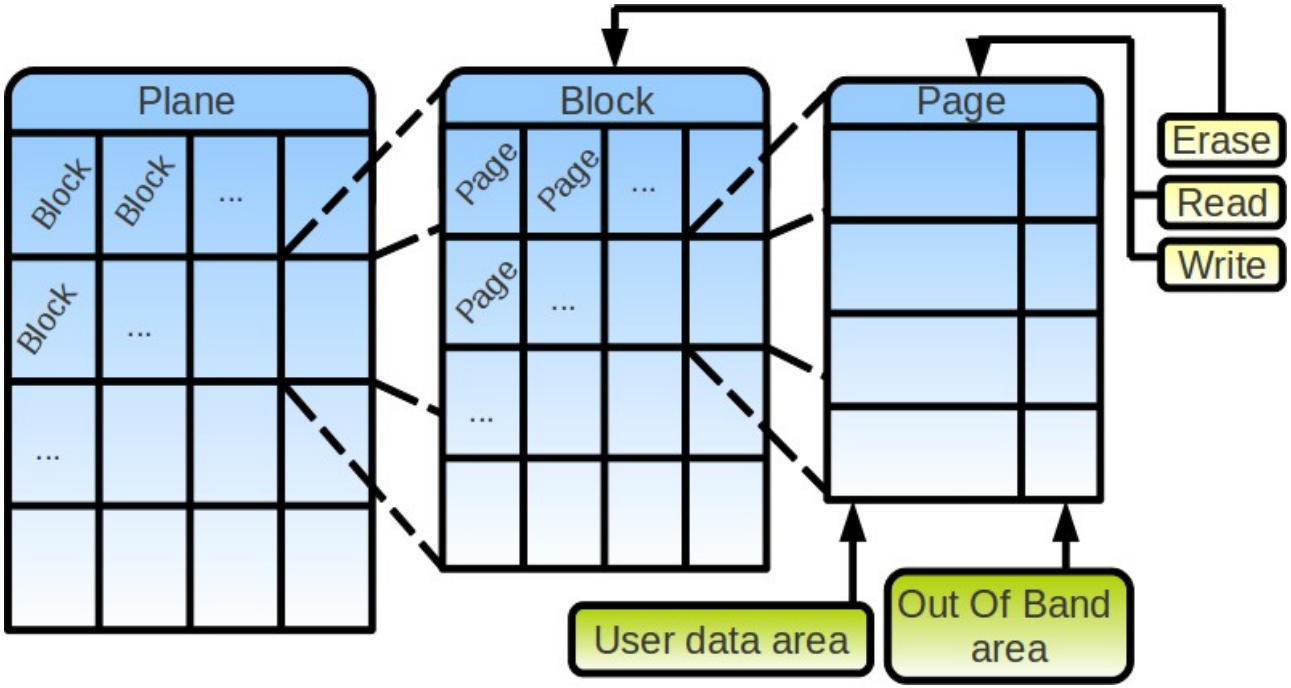
- Depend du #pages à écrire

  → PM < seuil < BM

Applications
Bonne Configuration de pilote de mémoire Flash

- Actions
  - Opérations E/R/W
  - E sur blocs (usure)
  - R/W sur pages
  - E avant W

Temps de réponse R/W



mapping des adresses

- Par page (PM) → coût RAM
- Par bloc (BM) → coût #E
- Hybride → %PM

Choix BM vs PM pour W

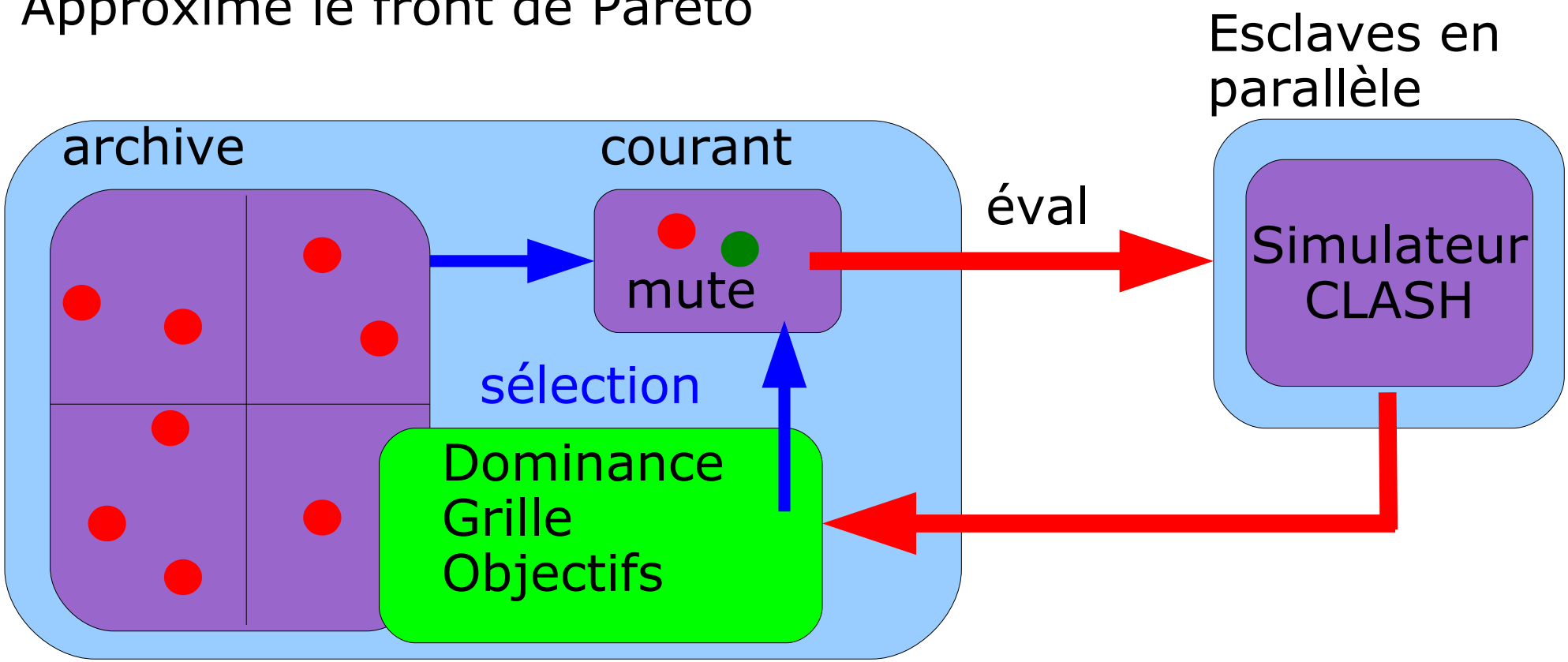
- Depend du #pages à écrire

→ PM < seuil < BM

Configuration de pilote de mémoire Flash
PAES en parallèle

Evaluation par simulateur de traces très longue
Pareto Archived Evolution Strategy en parallèle

Approxime le front de Pareto

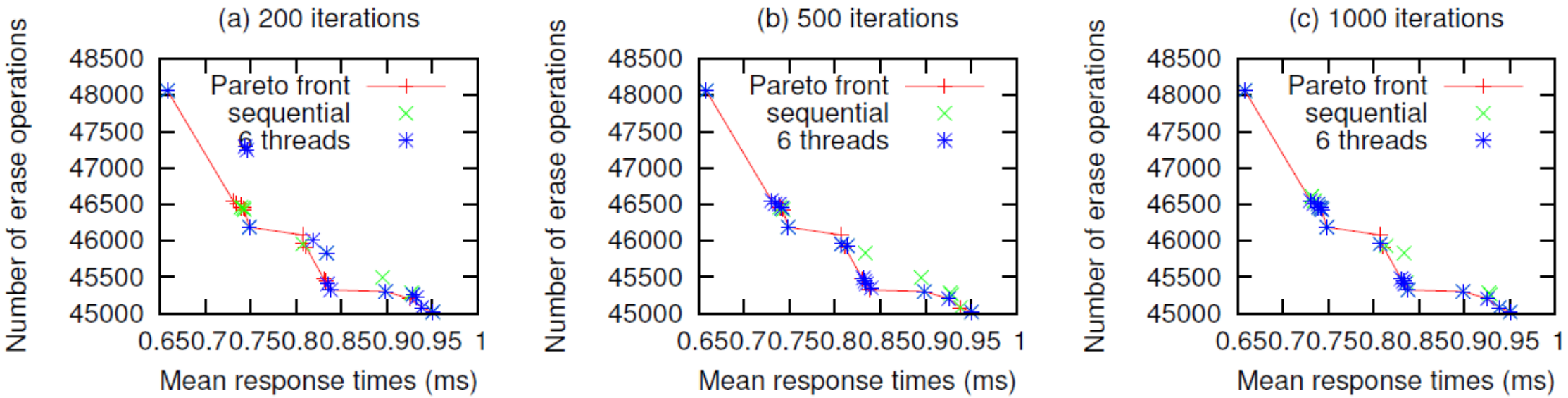


CLASH [Rubini,Boukhobza]

Configuration de pilote de mémoire Flash
PAES en parallèle
maitre/esclave asynchrone

Evaluation par simulateur de traces très longue
Pareto Archived Evolution Strategy en parallèle

6 esclaves : approche mieux le front de Pareto



temps d'exécution pour 100 itérations :

Method	set size	runtime (100 iterations)
Pareto Front	17	-
single PAES	2	577 s
1-slave PAES	2	536 s
6-slaves PAES	2	107 s

TD programmation lineaire
Allocation de ressources de calcul (ex. Amazon WS)

- Vente de temps de calcul sur des machines physiques
  - Machines physiques PM_j caractérisées par
    - Leur cout énergetique e_j ($ par unité de calcul)
    - Leur capacité de calcul c_j (en unités de calcul)
    - Leur cout fixe de démarrage f_j ($)
  - Les calculs correspondent aux lancements de VM
    - Nombre d'unités de calcul u_i nécessaire pour chaque VM_i
- Comment placer les VM_i sur les PM_j ?
  - Eventuellement, ressources limitées en PM de chaque type
  - Fonctions objectives possibles

TD programmation lineaire
Allocation de VM (Amazon WS)

- Proposer un codage des solutions

- Fct obj : **cout énergétique**
  - Démarage des machines
  - Couts unitaires
- Fct obj : **laxité**
  - Pas de migration obkigée en cas de variation de la charge
  - Min ou moyenne

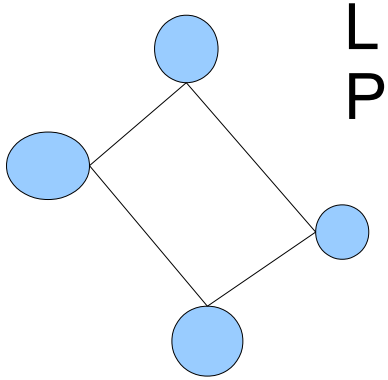
- Opérateurs de voisinage ? Mutations ? Crossovers ?

TD
PAES
TSP multi objectif

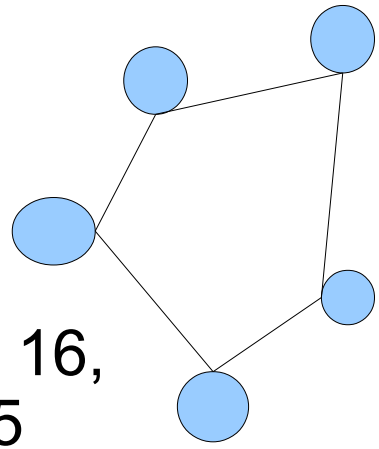
TSP SOO ≠ TSP MOO :
Toutes les villes
ne sont pas forcement incluses

- Données
  - G = (V, E)
  - Longueurs $v_i \rightarrow v_j$
  - Profits p_i
- MOO bi-objectif
  - min $\Sigma v_i \rightarrow v_j$ *vs* max Σp_i

    le sommet v_1 doit être inclus



L = 12,
P=4

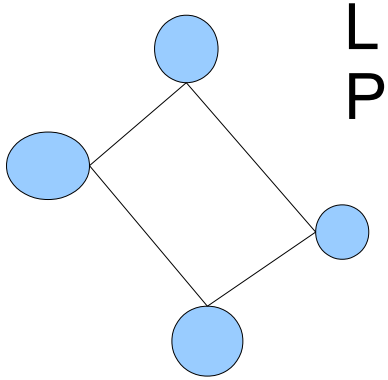


L = 16,
P=5

- Variantes (cf *Multi-objective Meta-heuristics for the Traveling Salesman Problem with Profits, M. Laguna.* DOI 10.1007/s10852-008-9080-2 – http://leeds-faculty.colorado.edu/glover/TSP%20-%20Multi-objective%20Metaheuristics%20w%20Jozefowiez%20and%20Laguna.pdf)
  - Borne sur profit min
  - Et/ou Borne sur longueur max

TD/TP
PAES
TSP multi objectif

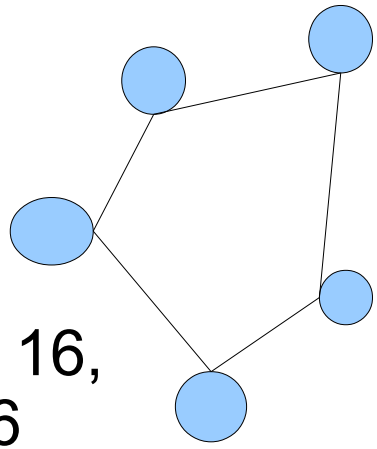
- Comment avoir un problème bi objectif à minimiser ?
  → solution simple puis pb MOO



L = 12,
P=4

- Proposer un codage des solutions

- Proposer une procédure de mutation (opérations possibles?)
- Proposer un algorithme de recherche locale

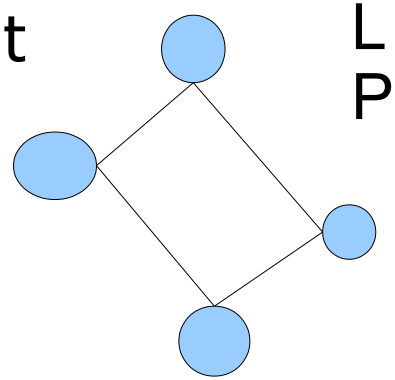


L = 16,
P=6

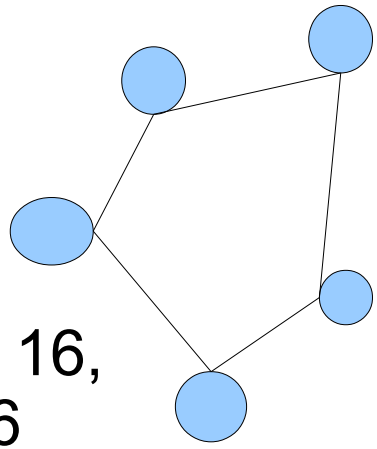
- Lire la doc PAES fournie (paescloud.c), quels pourraient être les paramètres d'appel ?

TD/TP
PAES vs NSGA2
TSP multi objectif

- D'après la doc fournie (Readme), que doit être développé pour NSGA2 ?
- Qu'est ce qui peut être réutilisable de la version PAES avec NSGA2 ?
- Proposer une procédure de mutation
- A quoi servent les pénalités ? Quelle métrique proposez vous ?
- Comment traiter les différentes versions de TSP MOO ?



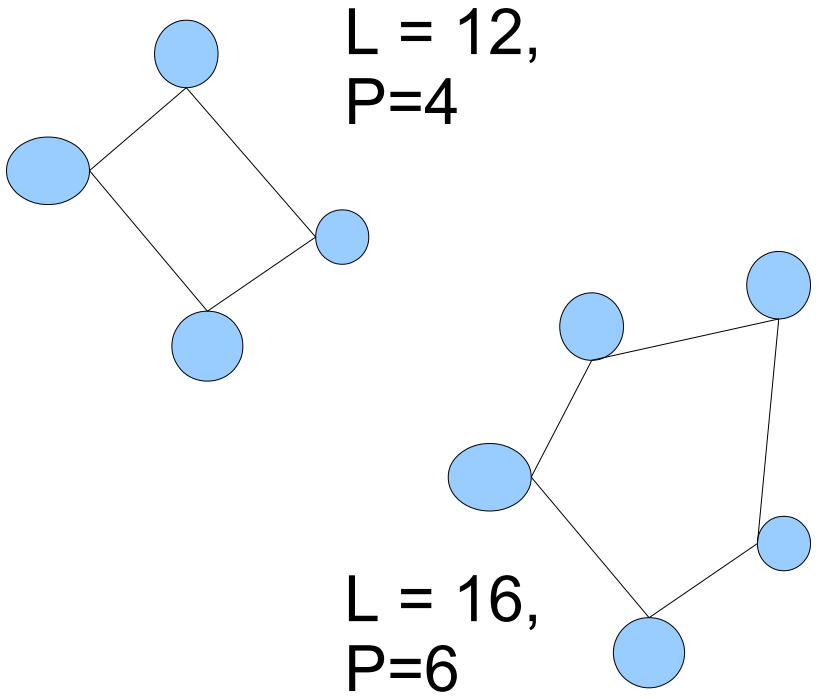
L = 12,
P=4



L = 16,
P=6

TP 1
PAES
TSP multi objectif

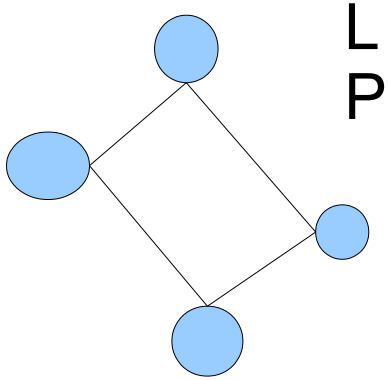
- Prendre en main le code C PAES
  → exemple de paescloud.c
- Regarder les fonctions utilitaires liées au TSP MOO (lecture des données)
  → gapaes.c
- Coder la procédure de mutation
  → paestsp.c à partir de paescloud.c



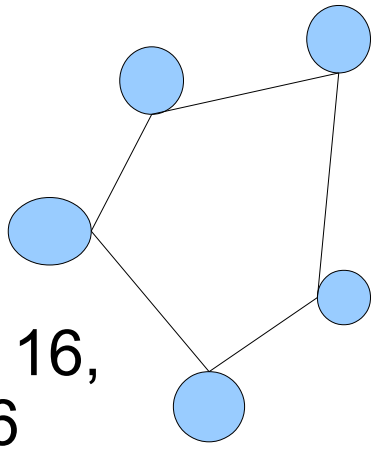
- Coder la fonction d'évaluation
- Tester
- Traiter les différentes versions de TSP MOO

TP 2
PAES vs NSGA 2
TSP multi objectif

- Ajouter une recherche locale
- Tester l'algorithme, dessiner le front de Pareto (archive) obtenu avec gnuplot
- Calculer l'hyper-volume (source hv)
- Traiter les différentes versions de TSP MOO
- Réaliser une implantation NSGA2 comparer les 2



L = 12,
P=4



L = 16,
P=6