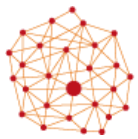


LocalSolver

LocalSolver 4.0

Programmation Mathématique Hybride

Thierry Benoist **Julien Darlay** Bertrand Estellon
Frédéric Gardi Romain Megel

www.localsolver.com

Qui sommes nous?

Groupe industriel diversifié présent dans
la construction, les télécoms et les médias

http://www.bouygues.com

Innovation24

Filiale optimisation de Bouygues
15 ans d'expérience dans la R.O.

http://www.innovation24.fr

LocalSolver

Solveur de programmation mathématique
pour l'optimisation combinatoire et continue

http://www.localsolver.com

LocalSolver

Solveur pour l'optimisation combinatoire et continue

- Permet de traiter des problèmes de grande taille
- Fournit de bonnes solutions en des temps courts
- Formalisme de modélisation simple
  - APIs en C++, Java, .NET
  - Langage de modélisation innovant (LSP)

Solveur basé sur la recherche locale

- Mouvements basés sur l'hypergraphe décisions/contraintes
- Évaluation incrémentale : millions de mouvements par minute
- Recuit simulé adaptatif, randomisé, parallélisé, avec restart

LocalSolver

Solveur de programmation mathématique

- **Pour l'optimisation combinatoire**
- Pour l'optimisation numérique
- Pour l'optimisation en variables mixtes

- **Fournit des solutions (bornes supérieures)**
- Fournit des bornes inférieures
- Preuve d'infaisabilité, écart ou preuve d'optimalité

Idéal pour l'optimisation non-convexe à grande échelle

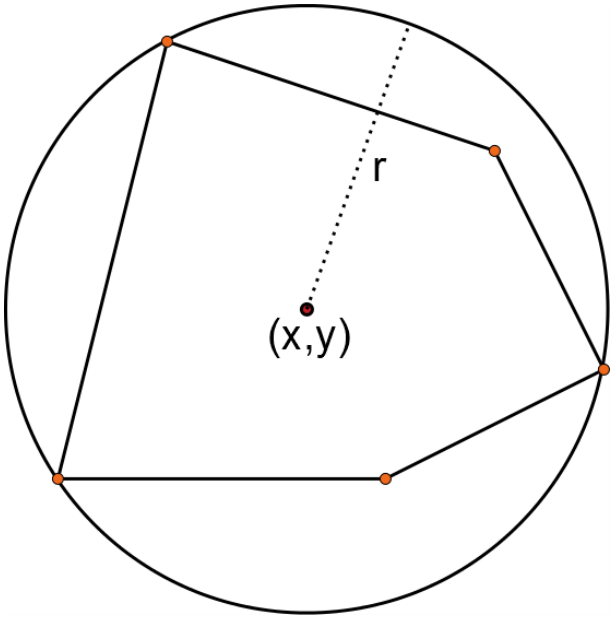
- Millions de variables combinatoires et/ou continues
- Contraintes et/ou objectifs non-convexes
- Temps de résolution courts

Exemples

Optimisation numérique

Smallest Circle

- Trouver le cercle de rayon minimal contenant un ensemble de points
- Deux variables de décisions continues ***x*** et ***y***
- Le rayon ***r*** : une expression déduite des décisions
- Modèle quadratique simple et naturel



Décision continue

Expression quadratique

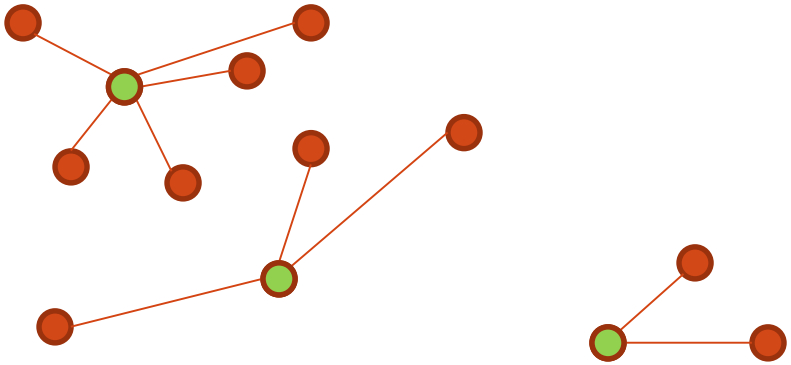
```
x <- float(minX, maxX);
y <- float(minY, maxY);

r2 <- max[i in 1..n] (pow(x-coordX[i],2) + pow(y-coordY[i],2));

minimize sqrt(r2);
```

LocalSolver

Optimisation combinatoire

P-Median

- Trouver un sous-ensemble de P éléments dans un ensemble de N
- Minimiser la somme des distances entre chaque élément et l'élément le plus proche dans P



```
function model() {
     x[1..N] <- bool();
     constraint sum[i in 1..N] (x[i]) == P;

     minDistance[i in 1..N] <- min[j in 1..N]  (x[j] ? distance[i][j] : +inf);
     minimize sum[i in 1..N] (minDistance[i]);
}
```

LocalSolver

Recherche Locale

Recherche Locale

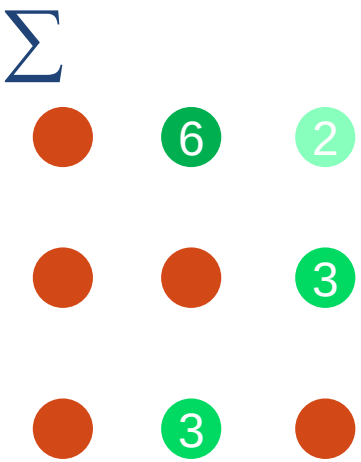
Idées principales en optimisation combinatoire

- Modification séquentielle d'un petit nombre de variables de décision
- Maintenir la faisabilité de la solution courante
- Évaluation incrémentale, si possible en temps constant

→ Connu sous le nom « *direct search* » en optimisation continue

Une architecture en trois couches

- Mouvements basé sur le modèle mathématique
- Évaluation incrémentale des solutions
- Heuristique qui pilote la recherche



Voisinages

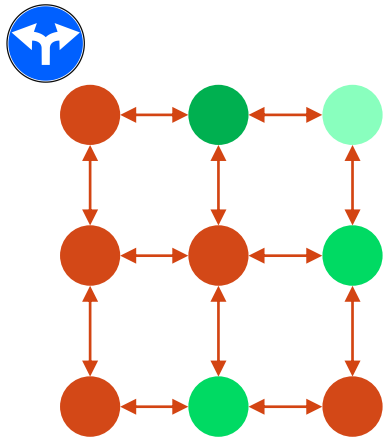
Mouvement classique en optimisation combinatoire: “k-flips”

- Peut amener à des solutions infaisables en pratique
- Si la faisabilité est dure à obtenir: convergence lente

LocalSolver maintient la faisabilité

- « *Destroy & repair* »
- Chaines d’éjection sur le graphe des contraintes
- Utilisation des structures combinatoires

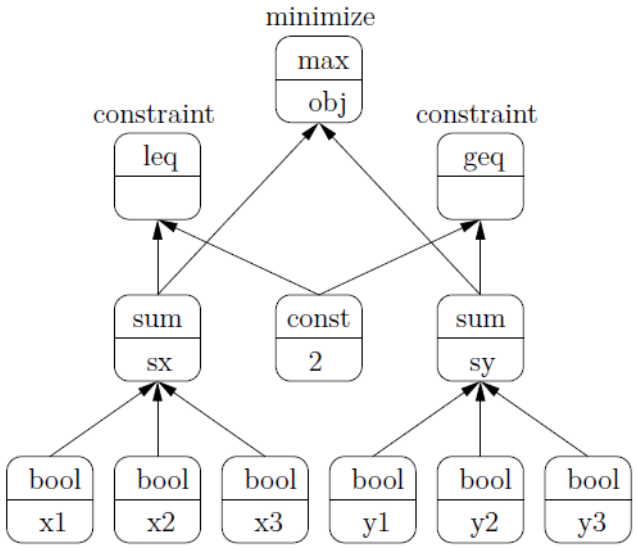
```
...
x[1..N] <- bool();
constraint sum[i in 1..N] (x[i]) == P;
...
```



LocalSolver

Évaluation rapide

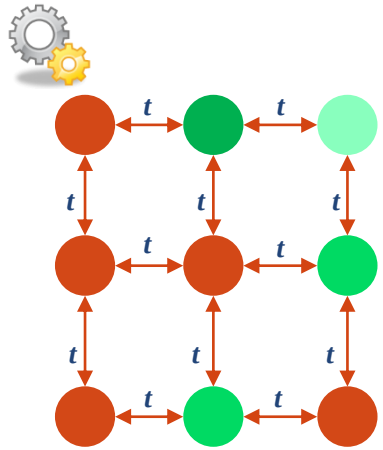
```
x1 <- bool();
x2 <- bool();
x3 <- bool();
y1 <- bool();
y2 <- bool();
y3 <- bool();
sx <- sum(x1, x2, x3);
sy <- sum(y1, y2, y3);
constraint leq(sx, 2);
constraint geq(sy, 2);
obj <- max(sx, sy);
minimize obj;
```



Évaluation incrémentale

- Propagation "*Lazy*" dans le DAG des expressions
- Utilise les invariants des opérateurs

⟶ Millions de mouvements par minute



Heuristique

Apprentissage en ligne des mouvements

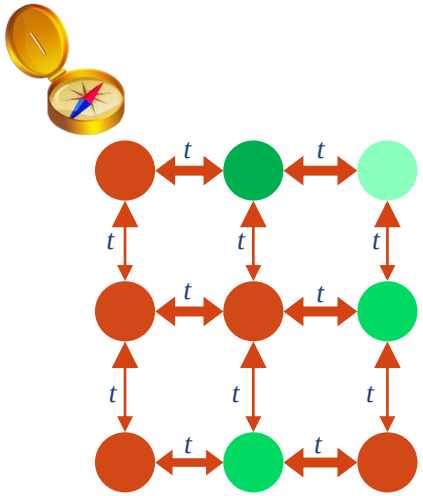
- Écarter les mouvements inefficaces sur un problème
- Favoriser les mouvements les plus pertinents

Recuit simulé adaptatif

- Gérer les objectifs non lisses
- Diversifier en acceptant des solutions dégradantes

« *Restart* » + recherches parallèles

- Échapper aux optima locaux
- Mieux couvrir l'espace de recherche



Benchmarks

Optimisation combinatoire

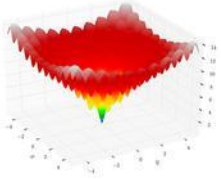
Car Sequencing : ordonnancer une ligne d'assemblage de véhicules

10 sec	100	200	300	400	500
Gurobi 5.5	140	274	X	429	513
LocalSolver 4.0	8	5	8	10	19
60 sec	**100**	**200**	**300**	**400**	**500**
Gurobi 5.5	3	66	1	356	513
LocalSolver 4.0	6	4	3	5	6
600 sec	**100**	**200**	**300**	**400**	**500**
Gurobi 5.5	3	2	*0	1	20
LocalSolver 4.0	4	*0	*0	2	*0

Optimisation non-convexe sans contrainte

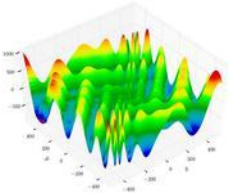
Solutions quasi optimales en quelques secondes sur une cinquantaine de paysages artificiels de la littérature

Oldenhuis (2009). Test functions for global optimization algorithms. Matlab



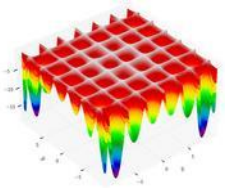
$$f(x,y) = -20\exp\left(-0.2\sqrt{0.5\left(x^2+y^2\right)}\right) - \exp\left(0.5\left(\cos\left(2\pi x\right)+\cos\left(2\pi y\right)\right)\right) + 20 + e.$$

gap (%) < 10^{-6}



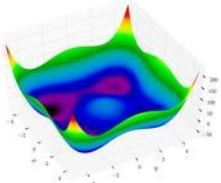
$$f(x,y) = -\left(y+47\right)\sin\left(\sqrt{\left|y+\frac{x}{2}+47\right|}\right) - x\sin\left(\sqrt{\left|x-\left(y+47\right)\right|}\right).$$

gap (%) < 10^{-4}



$$f(x,y) = -\left|\sin\left(x\right)\cos\left(y\right)\exp\left(\left|1-\frac{\sqrt{x^2+y^2}}{\pi}\right|\right)\right|.$$

gap (%) < 10^{-4}



$$f(\boldsymbol{x}) = \frac{\sum_{i=1}^{n} x_i^4 - 16x_i^2 + 5x_i}{2}.$$

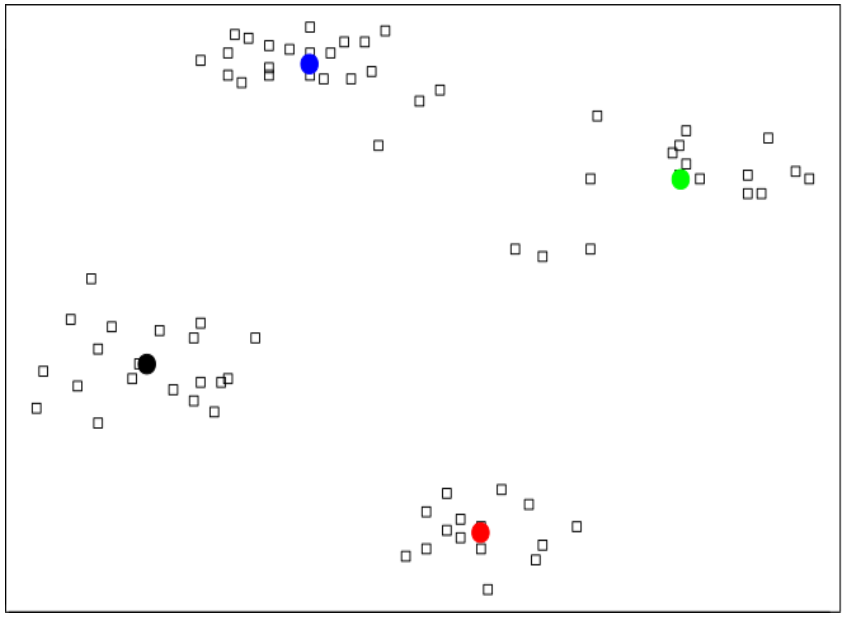
n = 10 → 10000

gap (%) < 10^{-6} → 10^{-1}

Optimisation combinatoire / continue

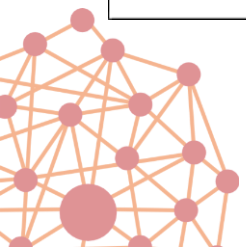
K-means

- Problème de « *machine learning* »
- NP-Difficile, Quadratique
- Solutions équivalentes comb. / cont.



Instance	k	OPT*	LS 4.0	GAP
ruspini	2	89337	89337,9	0,00%
	3	51063,4	51063,5	0,00%
	4	12881	12881,1	0,00%
	5	10126,7	10126,8	0,00%
	6	8575,41	8670,86	1,11%
	7	7126,2	7159,13	0,46%
	8	6149,64	6158,26	0,14%
	9	5181,64	5277,11	1,84%
	10	4446,28	4856,98	9,24%
iris	2	152,348	152,369	0,01%
	3	78,8514	78,9412	0,11%
	4	57,2285	57,3556	0,22%
	5	46,4462	46,5363	0,19%
	6	39,04	41,7964	7,06%
	7	34,2982	34,6489	1,02%
	8	29,9889	30,3029	1,05%
	9	27,7861	28,0667	1,01%
	10	25,834	26,0521	0,84%
glass	20	114,646	120,048	4,71%
	30	63,2478	74,1251	17,20%
	40	39,4983	58,3912	47,83%
	50	26,7675	52,4679	96,01%

*[Aloise et al. 2012]



LocalSolver

Optimisation en variables mixtes

Optimisation des vallées hydrauliques

- 4 barrages hydroélectriques avec 2 à 6 pompes par barrage
- 30 unités de production thermique
- Apports d'eau et prix de l'électricité variables dans le temps
- Horizon long : 8000 pas de temps

→ Système dynamique fortement non-linéaire, en variables mixtes, avec contraintes très couplantes, à grande échelle (1 million de décisions)

Cplex, Xpress, Gurobi : pas de solution après 3 h

LocalSolver : solution de qualité en une minute

Vers un solveur « complet »

Un solveur pour attaquer tout type de problèmes

- Optimisation combinatoire, continue, ou mixte
- Problèmes de petite ou grande taille
- Capable de prouver l'optimum ou l'infaisabilité
- Capable de passer à l'échelle de façon heuristique

Un solveur offrant le meilleur de toutes les techniques

- Recherche locale & directe
- Propagation de contraintes et inférence
- Programmation linéaire/en nombres entiers
- Programmation non-linéaire (convexe et non-convexe)
- Programmation dynamique
- Algorithmes spécifiques (chemins, flots, couplages, etc.)