Mathematical programming by Local Search

T. Benoist, J. Darlay, B. Estellon, F. Gardi, R.Megel

LocalSolver

LocalSolver

Who?

Innovation 24

BOUYGUES

Large industrial group with businesses in construction, telecom, media

www.bouygues.com

Innovation24

Operation Research subsidiary of the Bouygues group

www.innovation24.fr

LocalSolver

Flagship product of Innovation 24

www.localsolver.com

LocalSolver

Why?

Practical observations

What is the most powerful tool provided by OR today?
→ Mixed Integer Linear Programming (MIP)

- Simple and generic formalism
- Easy-to-use solvers: “model-and-run” approach
- Now an indispensable tool for practitioners
- Constraint Programming (CP) is following the way

What do practitioners when MIP/CP solvers are ineffective?
→ Local Search (LS)

- Core principle: improving the incumbent by exploring neighborhoods
- Provides quality solutions in minutes
- Extra costs (development, maintenance)

Our goals

A solver aligned with enterprise needs

- Provides high-quality solutions in seconds
- Scalable: tackles problems with millions of decisions
- Proves infeasibility or optimality when possible (best effort)

A solver aligned with practitioner needs

- « Model & Run »
  - Simple mathematical modeling formalism
  - Direct resolution: no need of complex tuning
- Full-version free trials with support
- Competitive pricing

http://www.localsolver.com/pricing.html

Free for academics

LocalSolver

Quick tour

Classical knapsack

8 items to pack in a sack: maximize the total value of items while not exceeding a total weight of 102 kg

```
function model() {
  // 0-1 decisions
  x_0 <- bool(); x_1 <- bool(); x_2 <- bool(); x_3 <- bool();
  x_4 <- bool(); x_5 <- bool(); x_6 <- bool(); x_7 <- bool();

  // weight constraint
  knapsackWeight <- 10*x_0+ 60*x_1+ 30*x_2+ 40*x_3+ 30*x_4+ 20*x_5+ 20*x_6+ 2*x_7;
  constraint knapsackWeight <= 102;

  // maximize value
  knapsackValue <- 1*x_0+ 10*x_1+ 15*x_2+ 40*x_3+ 60*x_4+ 90*x_5+ 100*x_6+ 15*x_7;
  maximize knapsackValue;
}
```

Binary decision variables

Integer intermediate variables

You write the model: nothing else to do!

declarative approach = model & run

LocalSolver

Multiobjective nonlinear knapsack

```
function model() {
  // 0-1 decisions
  x[0..7] <- bool();

 // weight constraint
  knapsackWeight <- 10*x[0]+ 60*x[1]+ 30*x[2]+ 40*x[3]+ 30*x[4]+ 20*x[5]+ 20*x[6]+ 2*x[7];
  constraint knapsackWeight <= 102;

  // maximize value
  knapsackValue <- 1*x[0]+ 10*x[1]+ 15*x[2]+ 40*x[3]+ 60*x[4]+ 90*x[5]+ 100*x[6]+ 15*x[7];
  maximize knapsackValue;

  // secondary objective: minimize product of minimum and maximum values
  knapsackMinValue <- min[i in 0..7](x[i] ? values[i]  : 1000);
  knapsackMaxValue <- max[i in 0..7](x[i] ? values[i]  : 0);
  knapsackProduct <- knapsackMinValue * knapsackMaxValue;
  minimize knapsackProduct;
}
```

Nonlinear operators: prod, min, max, and, or, if-then-else, …

Lexicographic objectives

Mathematical operators

Arithmetic			Logical	Relational
sum	prod	abs	not	==
min	max	dist	and	!=
div	mod	exp	or	<=
sqrt	log	pow	xor	>=
log	exp	tan	if	<
cos	sin	round	array + at	>
floor	ceil			

From LSP to APIs

```
function model() {
  // 0-1 decisi
  x[1..nbItems]

  // weight con
  knapsackWeig
  constraint kn

  // maximize
  knapsackValu
  maximize kn
}
```

C++

```
#include "localsolver.h"

using namespace localsolver;

int ma
{
  int
  int
```

Java

```
import localsolver.*;

public class Toy {

  publ
```

C#

```
using System;
using localsolver;

public class Toy
{
  static void Main()
  {
    int[] weights = {10, 60, 30, 40, 30, 20, 20, 2};
    int[] values = {1, 10, 15, 40, 60, 90, 100, 15};

    LocalSolver localsolver = new LocalSolver();
    LSModel model = localsolver.GetModel();

    // 0-1 decisions
    LSExpression[] x = new LSExpression[8];
    for (int i = 0; i < 8; i++)
      x[i] = model.CreateExpression(LSOperator.Bool);

    // knapsackWeight <- 10*x0 + 60*x1 + 30*x2 + 40*x3 + 30*x4 + 20*x5 + 20*x6 + 2*x7;
    LSExpression knapsackWeight = model.CreateExpression(LSOperator.Sum);
    for (int i = 0; i < 8; i++)
      knapsackWeight.AddOperand(model.CreateExpression(LSOperator.Prod, weights[i], x[i]));

    // knapsackWeight <= 102;
    model.AddConstraint(model.CreateExpression(LSOperator.Leq,

    // knapsackValue <- 1*x0 + 10*x1 + 15*x2 + 40*x3 + 60*x4 + 90*x5 + 100*x6 + 15*x7;
    LSExpression knapsackValue = model.CreateExpression(LSOperator.Sum);
    for (int i = 0; i < 8; i++)
      knapsackValue.AddOperand(model.CreateExpression(LSOperator.Prod, values[i], x[i]));

    // maximize knapsackValue;
    model.AddObjective(knapsackValue, LSObjectiveDirection.Maximize);

    // close the model before solving it
    model.Close();
    LSPhase phase = localsolver.CreatePhase();
    phase.SetTimeLimit(1);
    localsolver.Solve();
```

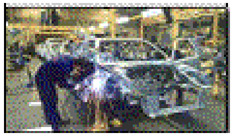
createExpression()

addOperand()

LocalSolver

Let's go inside

Car Sequencing



Scheduling cars on a production line

Objective = distributing options

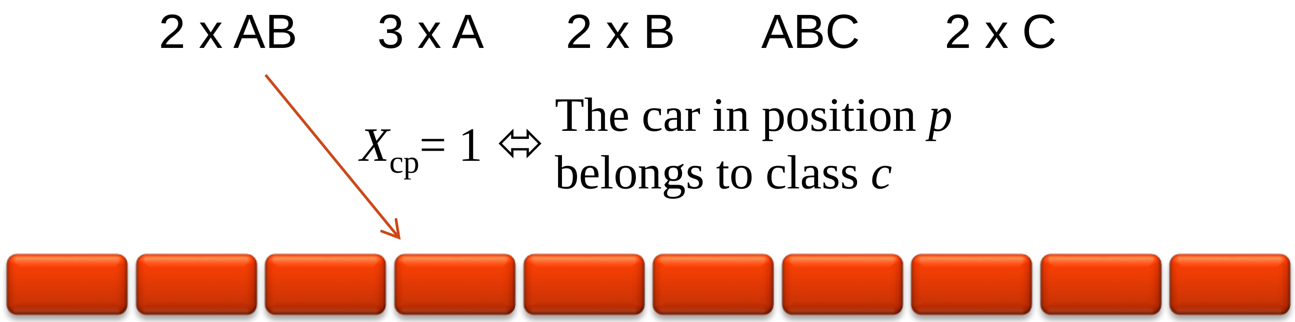
- E.g. : at most 2 sun-roofs in any sequence of 5 cars («P/Q»)
- measure: in each window of length 5, penalty based on overcapacities = max(n-2,0) with *n* the number of sun-roofs.

A *class* is a set of identical cars

- Her with 3 options A, B and C: AB is the class of cars featuring options A and B

AB	A	B	AB	A	ABC	C	A	B	C

Model




```
X[c in 1..nbClasses][p in 1..nbPositions] <- bool();

for[c in 1..nbClasses]
  constraint sum[p in 1..nbPositions](X[c][p]) == card[c];

for[p in 1..nbPositions]
  constraint sum[c in 1..nbClasses](X[c][p]) == 1;

op[o in 1..nbOptions][p in 1..nbPositions] <-
          or[c in 1..nbClasses : options[c][o]](X[c][p]);

nbVehicles[o in 1..nbOptions][j in 1..nbPositions-Q[o]+1] <-
              sum[k in 1..Q[o]](op[o][j+k-1]);

violations[o in 1..nbOptions][j in 1..nbPositions-Q[o]+1] <- max(nbVehicles[o][j] - P[o], 0 );

obj<- sum[o in 1..nbOptions][p in 1..nbPositions-Q[o]+1](violations[o][p]);
```

That's all!

Solving

How does LocalSolver solves this model ?

1. Find an initial solution (here a random assignment of cars)
2. Apply generic moves

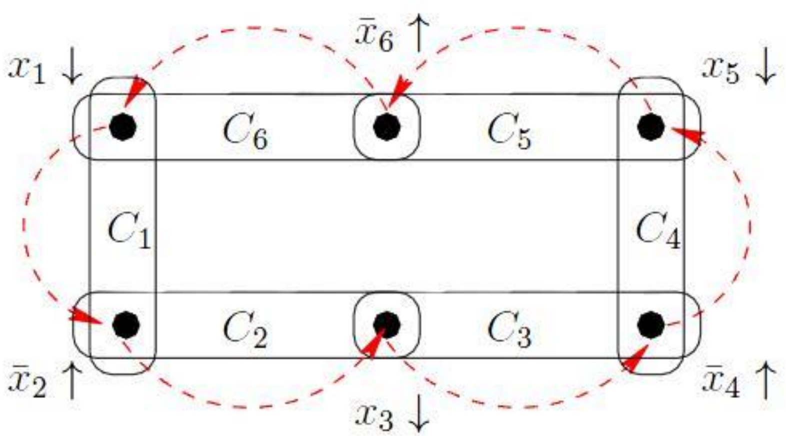
Small-neighborhood moves

Classical moves for Boolean Programming: "k-flips"

- Moves lead in majority to infeasible solutions
- Feasibility is hard to recover, implying a slow convergence
- Then no solver integrates an effective "pure local search" approach

Our moves tend to preserve the feasibility

- Can be viewed as a destroy-and-repair approach
- Can be viewed as ejection chains in the constraint hypergraph
- Can be specific to special combinatorial structures (when detected)



Solving

How does LocalSolver solves this model ?

1. Find an initial solution (here a random assignment of cars)
2. Apply generic moves

Exchanges (2 cars) — AB | A | B | AB | A | ABC | C | A | B | C

Exchanges involving 3 cars or more — AB | A | B | AB | A | ABC | C | A | B | C

Simple change — AB | A | B | AB | A | ABC | C | A | B | C

Violated constraint !
(2 cars at the same position)

Etc.

- Key points :
  - Simple changes will be eliminated after a few seconds since they fail systematically.
  - The global search strategy is a randomized simulated annealing (parameterized)
  - LocalSolver launches several concurrent search (the number of threads is a parameter)
  - Some moves will be focused on windows with overcapacities

For more details

T. Benoist, B. Estellon, F. Gardi, R. Megel, K. Nouioua.
LocalSolver 1.x: a black-box local-search solver for 0-1 programming.
4OR, A Quarterly Journal of Operations Research 9(3), pp. 299-316.

http://www.localsolver.com/technology.html

LocalSolver

Benchmarks

Car sequencing in Renault's plants

Some instances are public. This problem was submitted as ROADEF Challenge in 2005: http://challenge.roadef.org/2005/en

Example: instance 022_EP_ENP_RAF_S22_J1

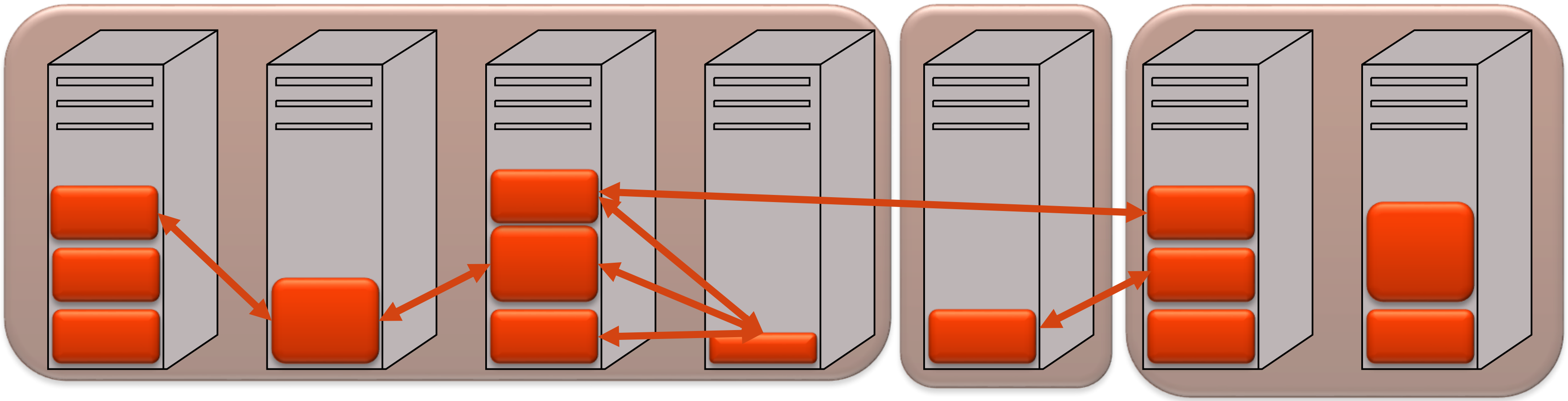
- Small instance: 80,000 variables, including 44,000 binary decisions
- State of the art: 3,109 obtained by a specific local search algorithm
- Best lower bound: 3,103

Results

- Gurobi 5.0: 3.116647e+07 in 10 min | 25,197 in 1 hour
- LocalSolver 3.0: 3,478 in 10 sec | 3,118 in 10 min

2012 ROADEF Challenge

Reassignment of processes to machines, with different kinds of constraints (mutual exclusion, resources, etc.)



More than 100 000 binary decisions

Only 1 day of work

LocalSolver qualified for final round (ranked 24/80)

Some results obtained on the hardest MIPLIB instances

- Lower objective is better
- 5 minutes time limit for both LocalSolver and MIP
- Models are not suitably modeled for LocalSolver

Minimization

Problem	Variables	LS 3.1	MIP
ds-big	4.9 M	9 844	62 520
ivu06-big	27.0 M	479	9 416
ivu52	2.5 M	4 907	16 880
mining	5.3 M	- 65 720 600	902 969 000
ns1853823	1.1 M	2 820 000	4 670 000
rmine14	1.3 M	- 3 470	-171
rmine21	6.7 M	- 3 658	- 185
rmine25	14.0 M	- 3 052	- 161

LocalSolver

Business cases

Business cases

- Supply Chain Optimization
- Workforce planning
- TV Media Planning
- Logistic clustering
- Street lighting maintenance planning
- Network deployment planning
- Energy optimization for tramway lines
- Placement of nuclear fuel assemblies in pools
- Painting shop scheduling
- Transportation of equipment

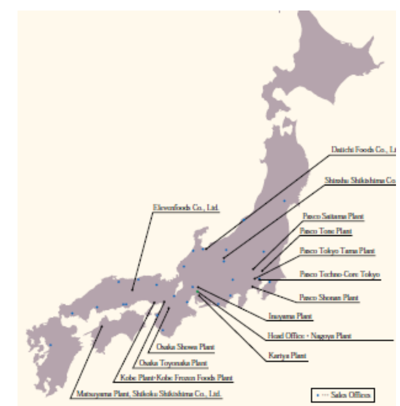
Supply Chain Optimization

Global Supply Chain

- Both production and logistics optimization
- More than 10 factories,
  each with several production lines
- Large number of stores and distribution centers

A challenging context for LocalSolver

- 20,00,000-variable model including 3 millions binaries
- A rich model involving setup costs, delivery times, packaging…
- A vain attempt to solve the problem with MIP solvers
- LocalSolver finds a high-quality solution in minutes



LocalSolver

ETDE

Tomorrow at 8:30 in this room

Long Term Planning with LocalSolver

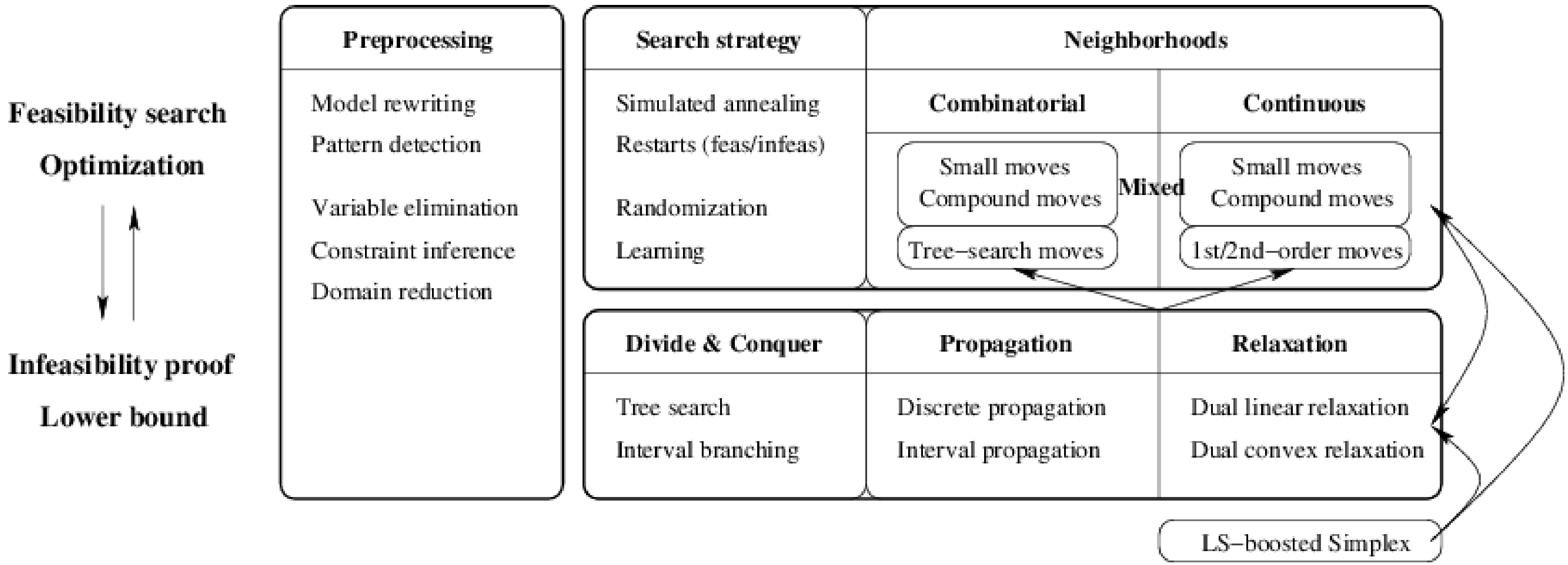
by Romain Megel

LocalSolver

Roadmap

Roadmap

Integrating MIP, CP, SAT techniques with LS into an all-in-one solver for large-scale mixed-variable non-convex optimization



LocalSolver

mathematical programming by local search

www.localsolver.com

Customers & Partners

ETDE

AIR LIQUIDE

ARMEE DE TERRE

BOUYGUES

altran

TF1 PUBLICITE

刃之砺
EdgeStone

Colas

FUJITSU

HEURISIS

Bouygues Telecom

EURODECISION
OPERATIONAL RESEARCH

mbi

MSI
Management Solutions Integration

DECIDE

Academic community

technische universität dortmund

東北大学 TOHOKU UNIVERSITY

BOĞAZİÇİ UNIVERSITY

THE UNIVERSITY OF EDINBURGH

Université catholique de Louvain UCL

UCC Coláiste na hOllscoile Corcaigh, Éire University College Cork, Ireland

UNSW THE UNIVERSITY OF NEW SOUTH WALES SYDNEY · CANBERRA · AUSTRALIA

DONGHUA UNIVERSITY

UPPSALA UNIVERSITET

UNIVERSITÀ DEGLI STUDI DI GENOVA

Universidad de Concepción Chile

POLITECNICO DI MILANO

Université de Valenciennes et du Hainaut-Cambrésis

université angers

UNIVERSITY of LIMERICK OLLSCOIL LUIMNIGH

The University of Nottingham

Universidade de São Paulo Brasil

UNI GRAZ

THE HONG KONG POLYTECHNIC UNIVERSITY 香港理工大學

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN

uff Universidade Federal Fluminense

Growing community!

TARBIAT MODARES UNIVERSITY

uco
UNIVERSITÉ CATHOLIQUE DE L'OUEST

EISTI
Ecole Internationale des Sciences du Traitement de l'Information

ONERA
THE FRENCH AEROSPACE LAB

THE UNIVERSITY OF TOKYO

ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΗΠΕΙΡΟΥ
EPIRUS INSTITUTE OF TECHNOLOGY
ARTA , ZIP CODE : 47100 , TEL NUMBER : +30 26810 50000

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
UPC

UNIVERSITÉ DE NANTES

CSIRO

भारतीय प्रौद्योगिकी संस्थान खड़गपुर
Indian Institute of Technology Kharagpur

UNIVERSITÄT DUISBURG ESSEN

KING SAUD UNIVERSITY

ENSTA ParisTech

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

IFSTTAR

STUDIORUM UNIVERSITAS FLORENTINA

universität wien

Universidad Tecnológica de Bolívar

JFLI
Japanese-French Laboratory for Informatics

LUX MEA LEX
SIGILLUM UNIVERSITATIS DAKARENSIS

DAUPHINE
UNIVERSITÉ PARIS

Linköping University

Grenoble INP

LocalSolver

mathematical programming by local search

www.localsolver.com