Beyond MILP:
Hexaly, a Hybrid Optimization Solver

Fred Gardi

Founder & CEO
fgardi@hexaly.com

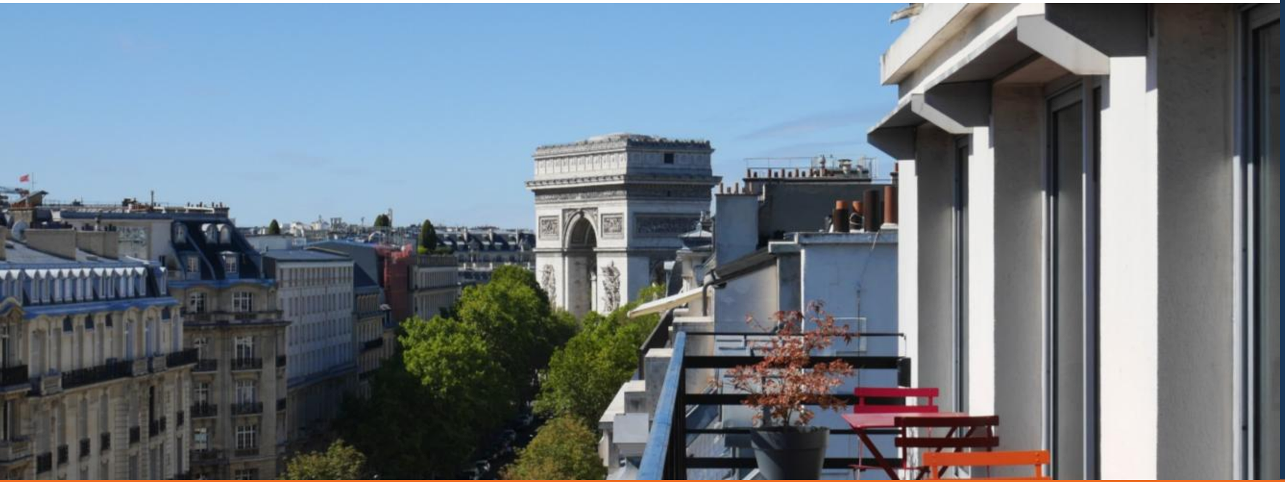
EWO Seminar, Carnegie Mellon University
May 7, 2026

www.hexaly.com

hexaly

Mathematical optimization platform

- Powerful, global optimization solver
  used by Amazon, FedEx, Procter & Gamble, Starbucks, Airbus, Renault, Bosch, …
- Low-code studio for math optimization



- Built a reputation for being super fast at solving Routing, Scheduling, Packing, …
- Bootstrapped in 2012
  Now 60 people, including 50 in R&D
- 400 client companies in 30 countries
  Fast-growing: x2 in the last 3 years
- Offices in Brooklyn, NY, and Paris, France

400 companies trust us

Complete platform for building and deploying mathematical optimization applications

Hexaly Optimizer
Global optimization solver

Hexaly Modeler
Modeling language for optimization

Hexaly Studio
Low-code studio for optimization

Hexaly Cloud
Optimization as a service

All free for academics and students!

Hexaly Optimizer

Hexaly Model

Python

Java

C#

C++

Hexaly Modeler

Solutions

Bounds

Hexaly Optimizer

- Delivers solutions and bounds, and so optimality gaps, for all objectives
- Provides optimality or inconsistency proofs

Beyond MILP:
The future is hybrid

What has guided us since 2012

Ambitious, original vision

John N. Hooker (2007)

“Good and Bad Futures for Constraint Programming (and Operations Research)”
Constraint Programming Letters 1, pp. 21-32

“Since modeling is the master and computation the servant, no computational method should presume to have its own solver.

This means there should be no CP solvers, no MIP solvers, and no SAT solvers. All of these techniques should be available in a single system to solve the model at hand.

They should seamlessly combine to exploit the problem structure. Exact methods should evolve gracefully into inexact and heuristic methods as the problem scales up.”

Hexaly Optimizer

Hybridizes and unifies all modeling paradigms into one solver:

- Mixed-Integer Programming (MIP)
- Non-Linear Programming (NLP)
- Constraint Programming (CP)
- Black-Box Optimization (BBO)
- Multi-Objective Optimization (MOO)

Adds new, CP-inspired concepts for combinatorial modeling:

- Sets and lists (= permutations)
- Intervals and optional intervals
- Map-reduce operators on variadic collections + lambda functions
- Global combinatorial operators over a set of collections

Hexaly Optimizer

Nonlinear, logical, relational algebra
+ set, list, interval algebra + black-box functions
+ multi-objective optimization

Decisional	Arithmetical			Logical	Relational	Set, list, interval	
bool	sum (+)	sub (-)	prod (*)	not (!)	eq (==)	**count**	**find**
float	div (/)	min	max	and (&&)	neq (!=)	**contains**	**distinct**
int	mod (%)	abs	dist	or (||)	geq (>=)	**at ([])**	**union**
set	floor	ceil	round	xor	leq (<=)	**indexOf**	**intersection**
list	pow (^)	exp	log	iif (? :)	gt (>)	**disjoint**	**start**
interval	sqrt	scalar	piecewise		lt (<)	**partition**	**end**
optionalInterval	cos	sin	tan			**cover**	**hull**

Where we are now: Hexaly 14.5

Hybridizes a myriad of exact and heuristic methods under the hood

LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	Derivative-Free Methods	Surrogate Modeling	Statistical Learning for Autotuning	

DD (2025)

Decision Diagram & Column Elimination techniques introduced in version 14.0 (2025)

LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	Derivative-Free Methods	Surrogate Modeling	Statistical Learning for Autotuning	

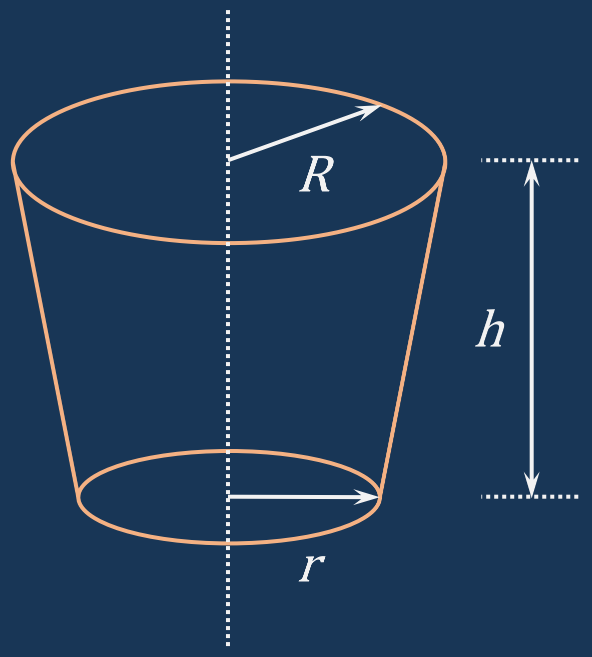
Mixed-Integer Non-Linear Optimization (MINLP)

Nonlinear modeling

Optimal Bucket Problem

Given a surface of metal $S = \pi$,

create a bucket of maximal volume



$$S = \pi r^2 + \pi(R + r)\sqrt{(R - r)^2 + h^2}$$

$$V = \frac{\pi h}{3}\,(R^2 + Rr + r^2)$$

Nonlinear modeling

Optimal Bucket Problem

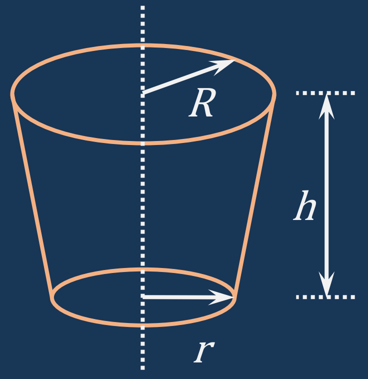
Here written using Hexaly Modeler (equivalent model can be written in Python, Java, C#, and C++)

```
function model() {
    PI = 3.14159265359;

    // Numerical decisions
    r <- float(0, 1e100);
    R <- float(0, 1e100);
    h <- float(0, 1e100);

    // Surface must not exceed the surface of the disc
    surface <- PI * pow(r, 2) + PI * (R + r)
            * sqrt(pow(R - r, 2) + pow(h, 2));
    constraint surface <= PI;

    // Maximize the volume
    volume <- PI * h / 3 * (pow(R, 2) + R * r + pow(r, 2));
    maximize volume;
}
```




```
Load optimal_bucket.lsp...
Run input...
Run model...
Run param...
Run solver...

Model:  expressions = 26, decisions = 3, constraints = 1,
objectives = 1
Param:  time limit = 2 sec, no iteration limit

[objective direction ]:     maximize

[  0 sec,       0 itr]:            0
[ optimality gap     ]:      100.00%
[  0 sec,   75503 itr]:     0.687095
[ optimality gap     ]:       < 0.01%

75503 iterations performed in 0 seconds

Optimal solution:
  obj    =     0.687095
  gap    =      < 0.01%
  bounds =     0.687174
```

Optimality proof

Hexaly Optimizer

Techniques at play on NLP/MINLP

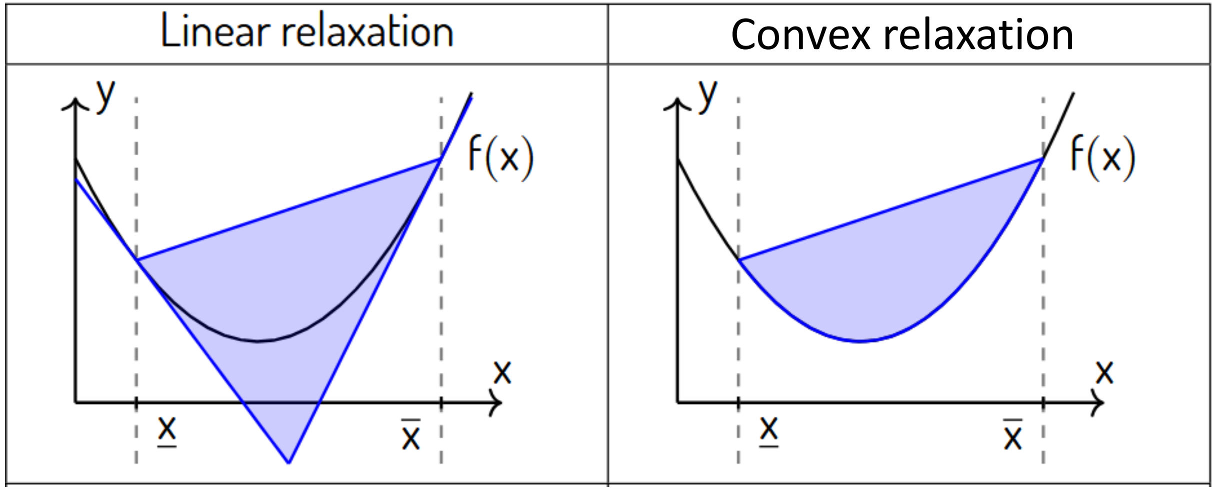
LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	Derivative-Free Methods	Surrogate Modeling	Statistical Learning for Autotuning	

(MI)NLP reformulation of the Optimal Bucket Problem

$$\min_{x \in \mathbb{R}^{15}} \frac{\pi}{3} x_{15}$$

$$\text{s.c.} \begin{cases} 0 \leq x_1 \leq 1 \\ 0 \leq x_2 \leq 1 \\ 0 \leq x_3 \leq 1 \\ x_{11} + x_{13} \leq 1 \\ x_8 = x_1 - x_2 \\ x_{10} = x_6 + x_9 \\ x_{12} = x_1 + x_2 \\ x_{14} = x_4 + x_5 + x_7 \\ x \in X \end{cases} \qquad X := \begin{cases} x_4 = x_1^2 \\ x_5 = x_2^2 \\ x_6 = x_3^2 \\ x_7 = x_1 x_2 \\ x_9 = x_8^2 \\ x_{11} = \sqrt{x_{10}} \\ x_{13} = x_{11} x_{12} \\ x_{15} = x_3 x_{14} \end{cases}$$

Outer approximation of nonlinear constraints



Relaxation solved with Simplex, IPM, or Augmented Lagrangian

Black-box optimization (BBO)

Black-box modeling

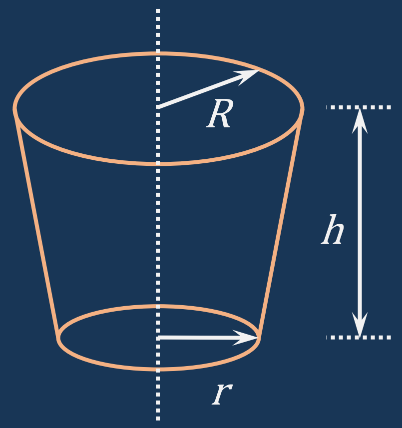
Optimal Bucket Problem

```
function model() {
    PI = 3.14159265359;

    // Numerical decisions
    r <- float(0, 1e100);
    R <- float(0, 1e100);
    h <- float(0, 1e100);

    // Surface must not exceed the surface of the disc
    constraint surface(r,R,h) <= PI;

    // Maximize the volume
    maximize volume(r,R,h);
}
```



Surface and volume are now given as hard-coded functions, given by the user

For example, these functions may be computed by numerical simulation

Many applications

Black-box functions may be:

- Monte-Carlo simulation
- Discrete-event simulation
- Agent-based simulation
- Numerical, physics-based simulation (Computational Fluid Dynamics)
- Machine learning (Neural Networks)
- Optimization: Hexaly may call Hexaly itself

Applications may be:

- Stochastic Optimization
- Robust Optimization
- Bilevel Optimization

Hexaly Optimizer

Techniques at play on BBO

LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	Derivative-Free Methods	Surrogate Modeling	Statistical Learning for Autotuning	

Hexaly Optimizer

If some BB functions are time-consuming: Surrogate Modeling

LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	**Derivative-Free Methods**	Surrogate Modeling	Statistical Learning for Autotuning	

Hexaly Optimizer

If some BB functions are time-consuming: Surrogate Modeling

LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	Derivative-Free Methods	**Surrogate Modeling**	Statistical Learning for Autotuning	

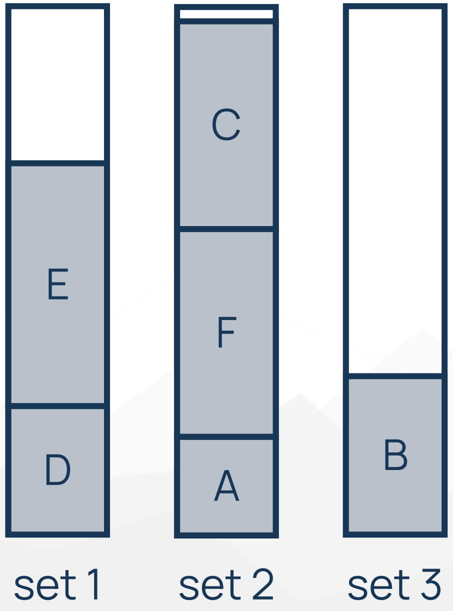
CP-inspired, innovative combinatorial modeling

Collection-based modeling

```
x <- set(N)
```

x is a **subset of {0, …, N-1}**

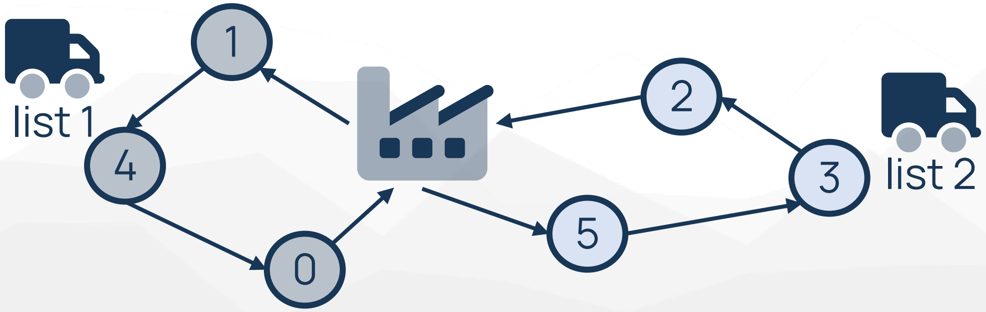
- Uniqueness of items
- **Variable** size




```
y <- list(N)
```

y is a **permutation of a subset of {0, …, N-1}**

- Uniqueness of items
- **Variable** size
- Items are **ordered**



Native combinatorial map-reduce

- Map-style element-wise transformation before aggregation
- Reduce-style aggregation over dynamic collections

$$\sum_{e \in s} e^2$$

```
s <- set(N);
sumOfSquaresOverS <- sum(s, e => e * e);
```

Traveling Salesman Problem

- N cities to visit
- Each city must be visited exactly one
- Minimize the total traveled distance

Traditional MILP models

With an **exponential number** of constraints

Minimise $\sum_{\substack{i,j \\ i \neq j}} c_{ij} x_{ij}$

Conventional Formulation (C) (Dantzig, Fulkerson and Johnson (1954))

$$\sum_{\substack{j \\ j \neq i}} x_{ij} = 1 \qquad \forall\ i \in N$$

$$\sum_{\substack{i \\ i \neq j}} x_{ij} = 1 \qquad \forall\ j \in N$$

$$\sum_{\substack{i,j \in M \\ i \neq j}} x_{ij} \leq |M| - 1 \qquad \forall\ M \subset N \text{ such that } \{1\} \notin M, |M| \geq 2$$

→ Requires an iterative procedure to manage subtour elimination constraints

Variant with **O(n²)** variables and constraints

SINGLE COMMODITY FLOW (**F1**) (Gavish and Graves (1978))

Both constraints are retained but we also introduce (continuous) variables:

y_{ij} = 'Flow' in an arc (i,j) $i \neq j$

and constraints:

$$y_{ij} \leq (n-1)x_{ij} \qquad \forall\ i,j \in N,\ i \neq j$$

$$\sum_{\substack{j \\ j \neq 1}} y_{1j} = n-1$$

$$\sum_{\substack{i \\ i \neq j}} y_{ij} - \sum_{\substack{k \\ i \neq k}} y_{jk} = 1 \qquad \forall\ j \in N - \{1\}$$

In Orman & Williams: *A survey of different integer programming formulations of the TSP*

Natural modeling

as a permutation

The Traveling Salesman Problem (TSP)

TSP: Given a list of cities and their pairwise distances, find a shortest possible tour that visits each city exactly once.

Objective: find a permutation $a_1,\ldots,a_n$ of the cities that minimizes

$$d(a_1,a_2)+d(a_2,a_3)+\ldots+ d(a_{n-1},a_n)+d(a_n,a_1)$$

where $d(i, j)$ is the distance between cities i and j

An optimal TSP tour through Germany's 15 largest cities

In Kenneth R. Rosen: *Permutations and Combinations*

Traveling Salesman Problem (TSP)

List-based modeling

```
function model() {
  // A list variable: cities[i]
  // is the index of the ith
  // city in the tour
  cities <- list(N);

  // All cities must be visited
  constraint count(cities) == N;

  // Minimize the total distance
  obj <- sum(1...N, i => distance[cities[i-1]][cities[i]])
        + distance[cities[N-1]][cities[0]];

  minimize obj;
}
```

$$\sum_{i=1}^{N} \text{distance}[c_{i-1}][c_i]$$

TSPLIB: rbg443.atsp

```
Load tsp.lsp...
Run input...
Run model...
Run param...
Run solver...

Model:  expressions = 495, decisions = 1, constraints = 1,
objectives = 1
Param:  time limit = 60 sec, no iteration limit

[objective direction ]:     minimize

[  0 sec,       0 itr]: No feasible solution found (infeas = 2)
[  1 sec,   40639 itr]:         3036
[  2 sec,  105606 itr]:         2835
[  3 sec,  161355 itr]:         2787
[  4 sec,  223548 itr]:         2751
[  5 sec,  342635 itr]:         2725
[  6 sec,  404510 itr]:         2720
[ optimality gap     ]:           0%

404510 iterations performed in 6 seconds

Optimal solution:
  obj    =         2720
  gap    =           0%
  bounds =         2720
```

Optimality proof

Capacitated Vehicle Routing Problem (VRP)

- One list variable for each vehicle of the fleet

```
tours[k in 0...nbTrucks] <- list(nbCustomers);
constraint partition(tours);
```

→ **Partition**: “global” combinatorial operator applied to an array of lists defined over the same domain

- Capacity constraint for each truck

```
sum(tours[k], i => demands[i]) <= capacity;
```

$$\sum_{c \in tours[k]} demands[c]$$

→ The number of terms in this sum is **variable**

Hexaly Optimizer

Techniques at play on collection-based models

LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	Column Generation & Branch-Cut-Price	Decision Diagrams & Column Elimination		
BBO	Derivative-Free Methods	Surrogate Modeling	Statistical Learning for Autotuning	

Hexaly Optimizer

Techniques at play on collection-based models

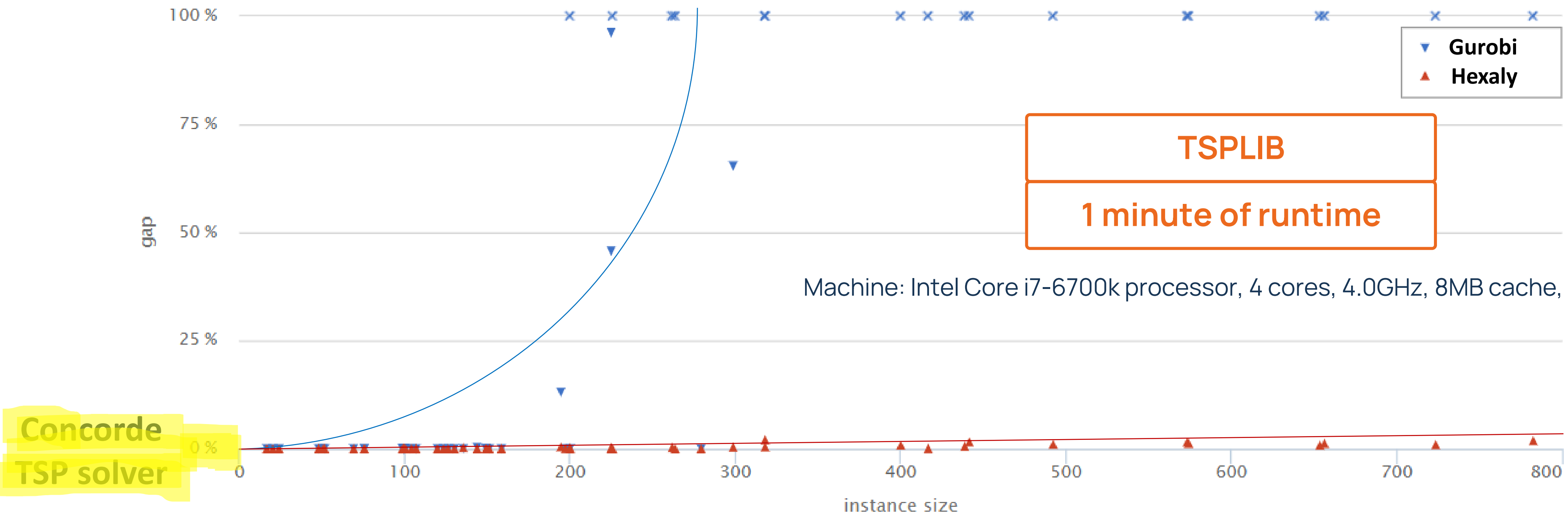
LP/QP/NLP	Presolve	Simplex	Interior-Point	Augmented Lagrangian
MILP/MINLP	Spatial Branch-and-Bound	Cutting Planes	Relaxation-based Primal Heuristics	
CP/SAT	Filtering	Constraint Propagation	Interval Methods	
LS	Greedy, GRASP, Population-based Methods	Local Search & Large Neighborhood Search		
BCP/DD	**Column Generation & Branch-Cut-Price**	**Decision Diagrams & Column Elimination**		
BBO	Derivative-Free Methods	Surrogate Modeling	Statistical Learning for Autotuning	

Traveling Salesman Problem

Benchmark against Concorde and MILP

hexaly.com/benchmark/hexaly-vs-gurobi-traveling-salesman-problem-tsp

- Gurobi fails to find solutions to TSP instances with 200 cities
- Hexaly finds near-optimal solutions (gap <1%) to TSP instances with **10,000 cities**



TSPLIB

1 minute of runtime

Machine: Intel Core i7-6700k processor, 4 cores, 4.0GHz, 8MB cache, 64GB RAM

More benchmarks available at hexaly.com/benchmarks

Benchmark

Job Shop Scheduling Problem (JSSP)

Hexaly vs CPO, OR-Tools, Gurobi, Cplex

Hexaly vs CP Optimizer, OR-Tools, Gurobi, Cplex on large-scale instances of the Job Shop Scheduling Problem (JSSP)

Read more →

Benchmark

New records for the Inventory Routing Problem (IRP)

Hexaly vs Research

Hexaly establishes new records for the Inventory Routing Problem (IRP)

Read more →

Benchmark

New records for the Resource-Constrained Project Scheduling Problem (RCPSP)

Hexaly vs Research

Hexaly breaks records for the Resource-Constrained Project Scheduling Problem (RCPSP)

Read more →

Benchmark

Car Sequencing Problem with Paint-Shop Batching Constraints

Hexaly vs Gurobi

Hexaly vs Gurobi on the Car Sequencing Problem with Paint-Shop Batching Constraints

Read more →

Benchmark

Flexible Job Shop Problem (FJSP)

Hexaly vs Gurobi

Hexaly vs Gurobi on the Flexible Job Shop Scheduling Problem (FJSP)

Read more →

Benchmark

Pickup and Delivery Problem with Time Windows (PDPTW)

Hexaly vs Google OR-Tools

Hexaly vs Google OR-Tools on the Pickup and Delivery Problem with Time Windows (PDPTW)

Read more →