The 8th International Conference "Distributed Computing and Grid-technologies in Science and Education"
10 – 14 September, 2018, Dubna, Russia

Discrete and Global Optimization in Everest Distributed Environment by Loosely Coupled Branch-and-Bound Solvers

Sergey Smirnov, ***Vladimir Voloshinov***

Center for Distributed Computing, http://distcomp.ru,
Institute for Information Transmission Problems (Kharkevich Institute),
Russian Academy of Sciences

Supported by the Russian Foundation for Basic Research, projects #18-07-01175, #18-07-00956

Outlines

- Briefly about Branch-and-Bound (BnB) method for MINLP
- Coarse-grained parallelization of BnB-solvers (vs fine-grained) by Domain Decomposition and/or Concurrent running
- DDBNB Everest application and experiments with Traveling Salesman Problem (MILP)
- DDBNB for Global optimization with examples of combinatorial geometry problems (Tammes, Thomson and Flat Torus Packing)
- Promising results with ParaSCIP solvers
- Conclusions and future plans.

Discrete Mathematical Programming (MILP & MINLP)

$$f_o(x) \to \min_x,$$
$$x = (x_B, x_C) \in Q, x_B \in \{0,1\}^{n_B}, x_C \in \mathbb{R}^{n_C}$$

(P)

$$Q = \left\{ f_i(x_B, x_C) \leqslant 0 (i \in \mathrm{I}), g_j(x_B, x_C) = 0 (j \in \mathrm{J}) \right\}$$
$$= \text{may be something else } \ldots$$

One of the algorithms is Branch-and-Bound (B&B) known from ~1960 (Land Al. H., Doig A. G. and etc.)

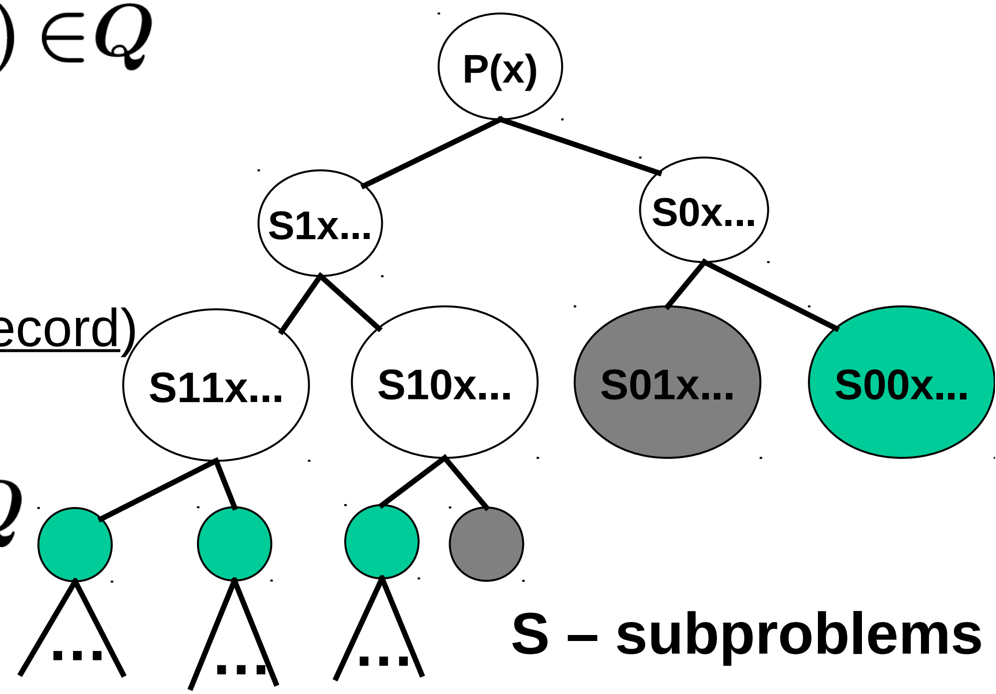
VERY briefly, B&B based on two interacting procedures:

Building the Search Tree	**Pruning Branch, Get Incumbents**
Recursive decomposition of feasible domain (***Q***), e.g. by fixing some x_B variables in accordance with some rules (branching)	Get lower bounds of obj. value for domain subsets; search feasible solutions $x' \in Q$ and keep the best ones, aka incumbents $f_o(x')$

Branch-and-bound for MI... problem (e.g. boolean)

General scheme of search tree traversing for problem (P)

$$f_o(x_B, x_C) \to \min_{(x_B, x_C)}, (x_B, x_C) \in Q$$



Current state of B&B (changed dynamically):

- list of nodes to be processed (green);

- upper-bound (**UB**) on MIN (aka <u>incumbent</u> | <u>record</u>)

$$\mathrm{UB} = f_o(x'_B, x'_C), (x'_B, x'_C) \in Q$$

Node operation:

1) **calculate lower-bound of S, LB(S), by relaxation** of boolean and/or non-convex constraints to, e.g. LP or convex MINLP;

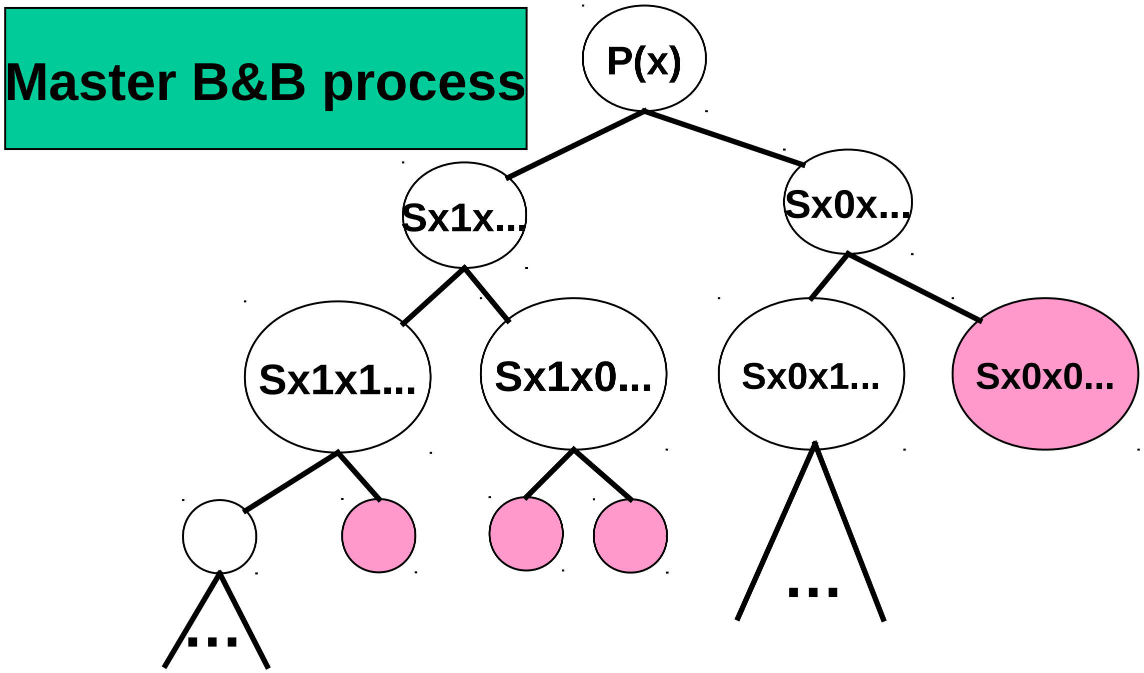
2) **if feasible vector was found** $\left(x''_B, x''_C\right) \in Q$ – update UB

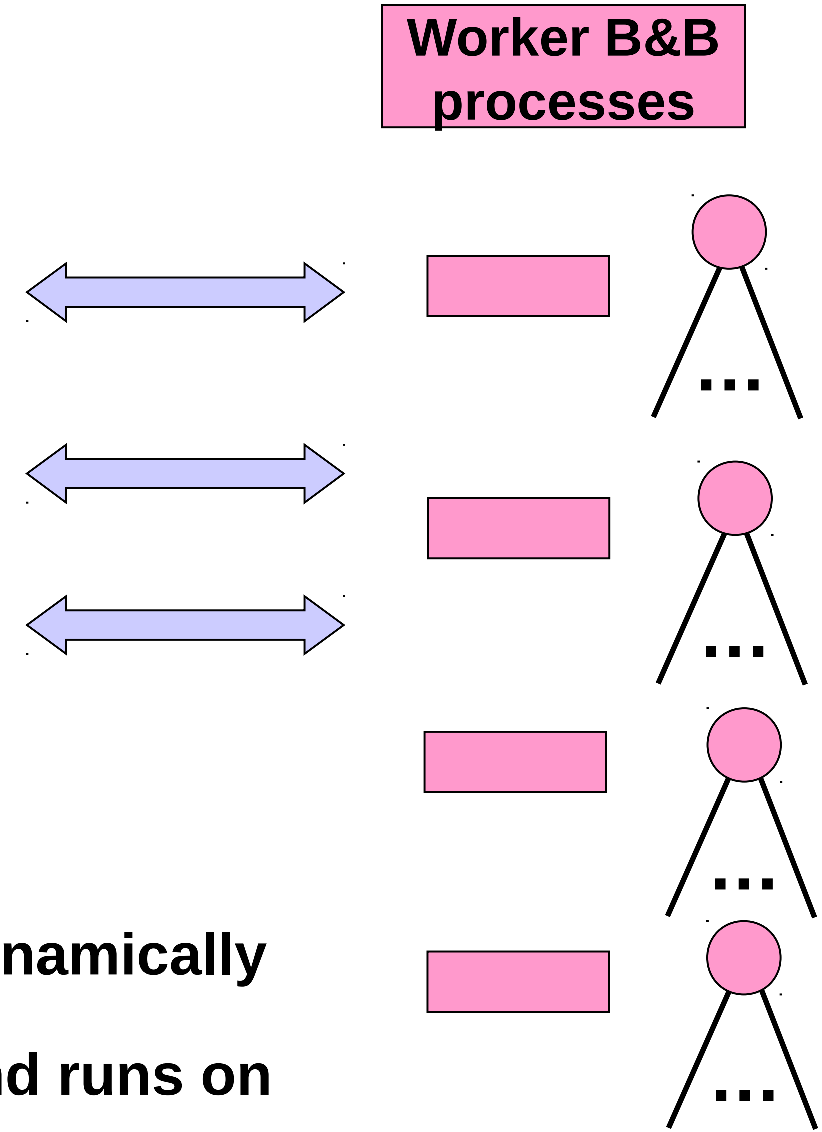
$$\mathrm{UB} := \min\{\mathrm{UB}, f_o(x''_B, x''_C)\}$$

3) **if LB(S) ⩾ UB – discard node** from the list (gray, “pruning” branch);

4) **select boolean variable (or add inequality with continuous variables) to decompose the node** and add new ones to the tree

Fine-grained parallelization of B&B (traditional approach)





Master & workers exchange with

<u>subtrees</u> and <u>incumbents.</u>

Subtrees (subproblems) are generated dynamically

Usually, this approach is based on MPI and runs on high-performance clusters.

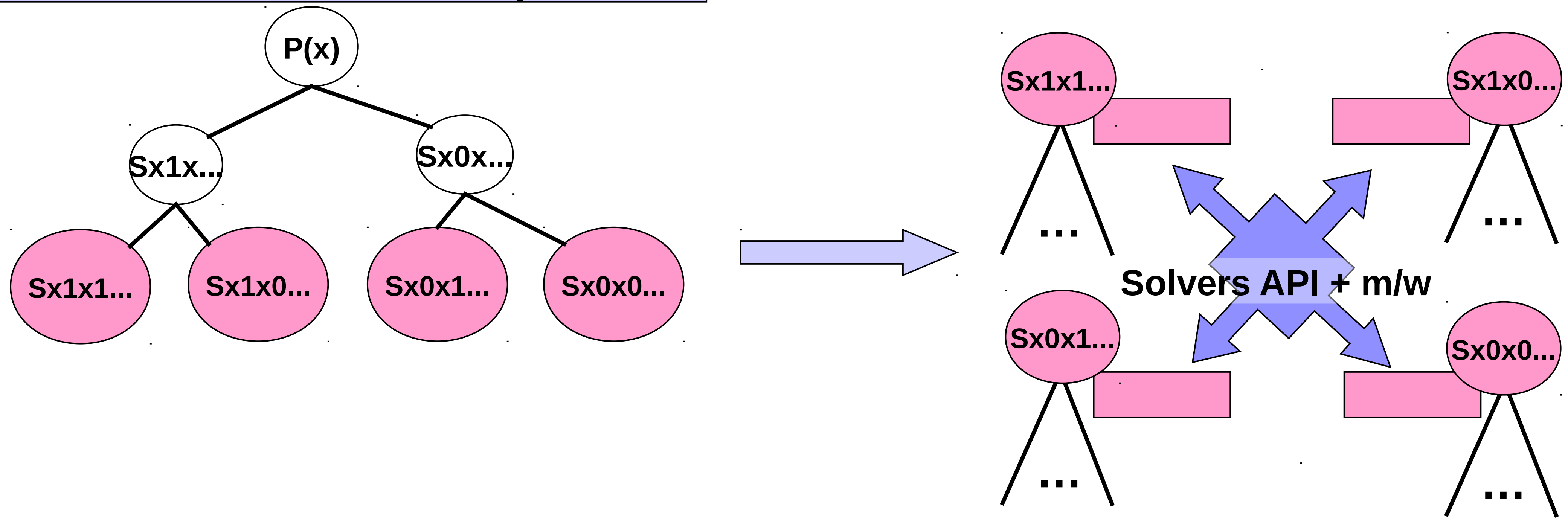
Rather intensive data flow.

Implementation requires low-level programming

Coarse-grained parallelization, DDBNB (DomainDecompositionB&B)

Preliminary Feasible Domain Decomposition

B&B solvers exchange incumbents only



The approach is not as popular as fine-grained one, but is much easier to implement via solvers' API and some "light-weight" middleware, e.g. Everest, ~~Erlang~~, Zeroc Ice, ZeroMQ etc.

Deciding on decomposition is crucial for speed-up. High-level tool to analyze the problem might be VERY useful ! E.g. AMPL or Pyomo.

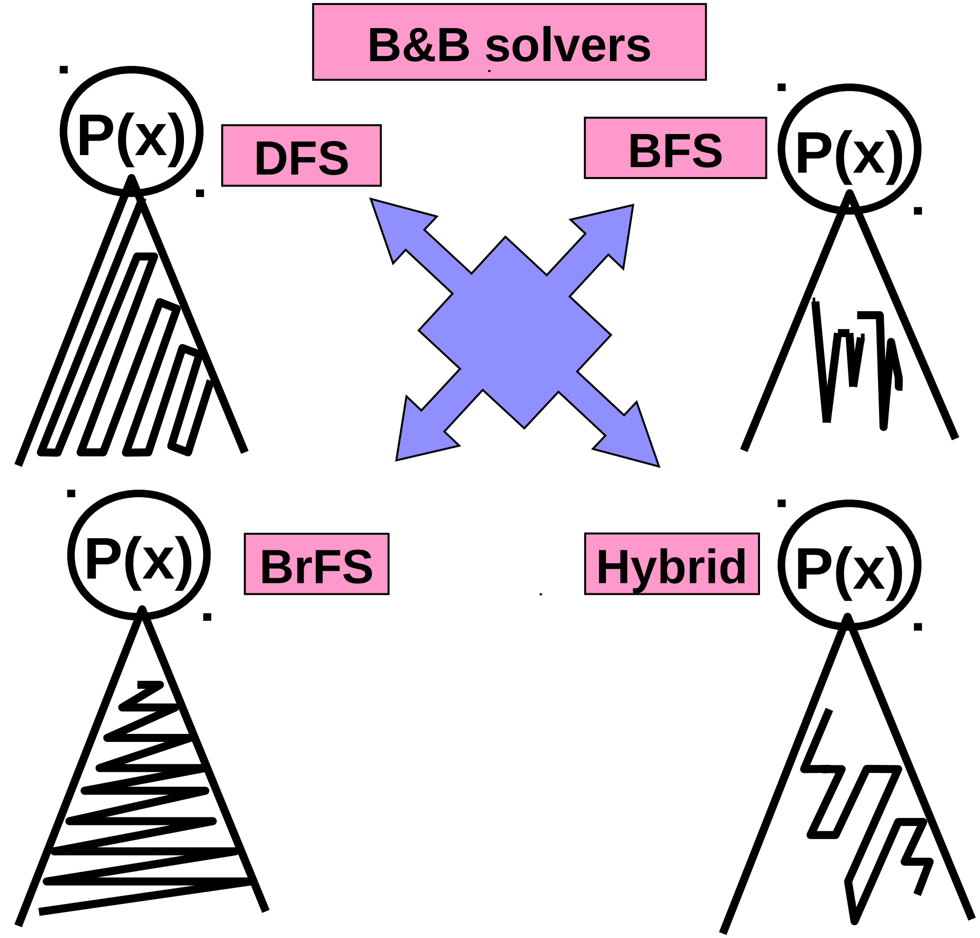
Concurrent (racing) parallelization

No decomposition. Concurrently running B&B with different settings on the same problem.

State-of-the-art solvers has a number of parameters (hundreds):

COIN-OR CBC >200 parameters

ZIB SCIP ~2000 parameters



SCIP, combination of only one *<u>nodeselection/childsel</u>* = {d,u,p,i,…} parameter, gives 20% speed-up. And even better!

DomainDecompositionB&B Everest Application (DD & Concurrent)

DDBNB, **https://github.com/distcomp/ddbnb**

Basic “ingredients”:

- High-level optimization modeling tools to perform decomposition:

  AMPL, A Modeling Language for Math. Program., ampl.com

  Pyomo (free, open source), PYthon Optimization MOdeling, **pyomo.org**, **AMPL-Compatible (!)**

- B&B solvers, AMPL-compatible, with open API:

  CBC, COIN-OR Branch-and-Cut, https://projects.coin-or.org/Cbc, **MILP**

  SCIP, Solve Constraint Integer Problem, **http://scip.zib.de**, **MIPolynomP!**

- Web-based platform, Everest, http://everest.distcomp.org provides:

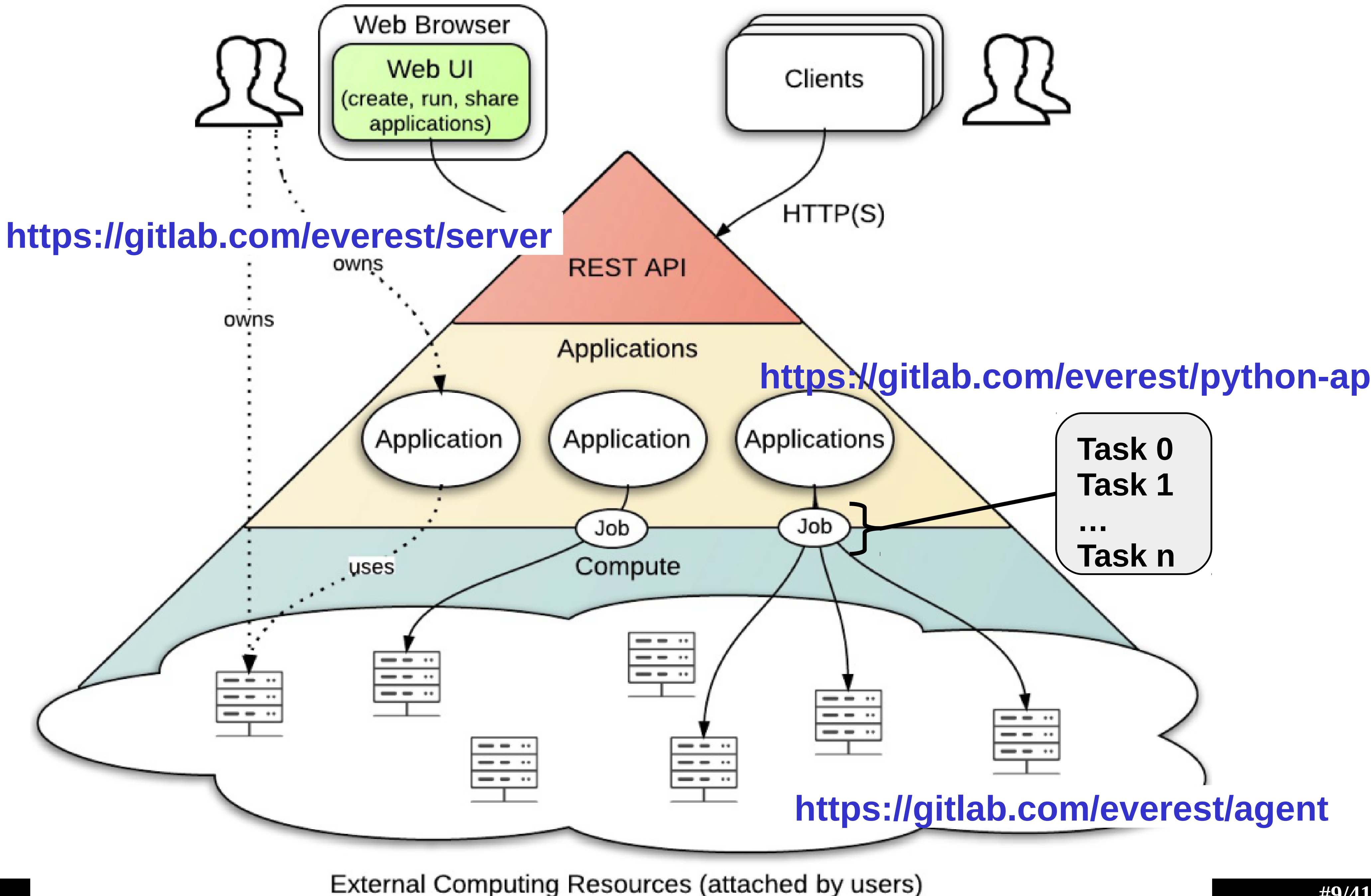
  integration of solvers installed on heterogeneous resources;

  generic service to run a pack of predefined tasks (subproblems);

  generic communication mechanism to exchange incumbents.

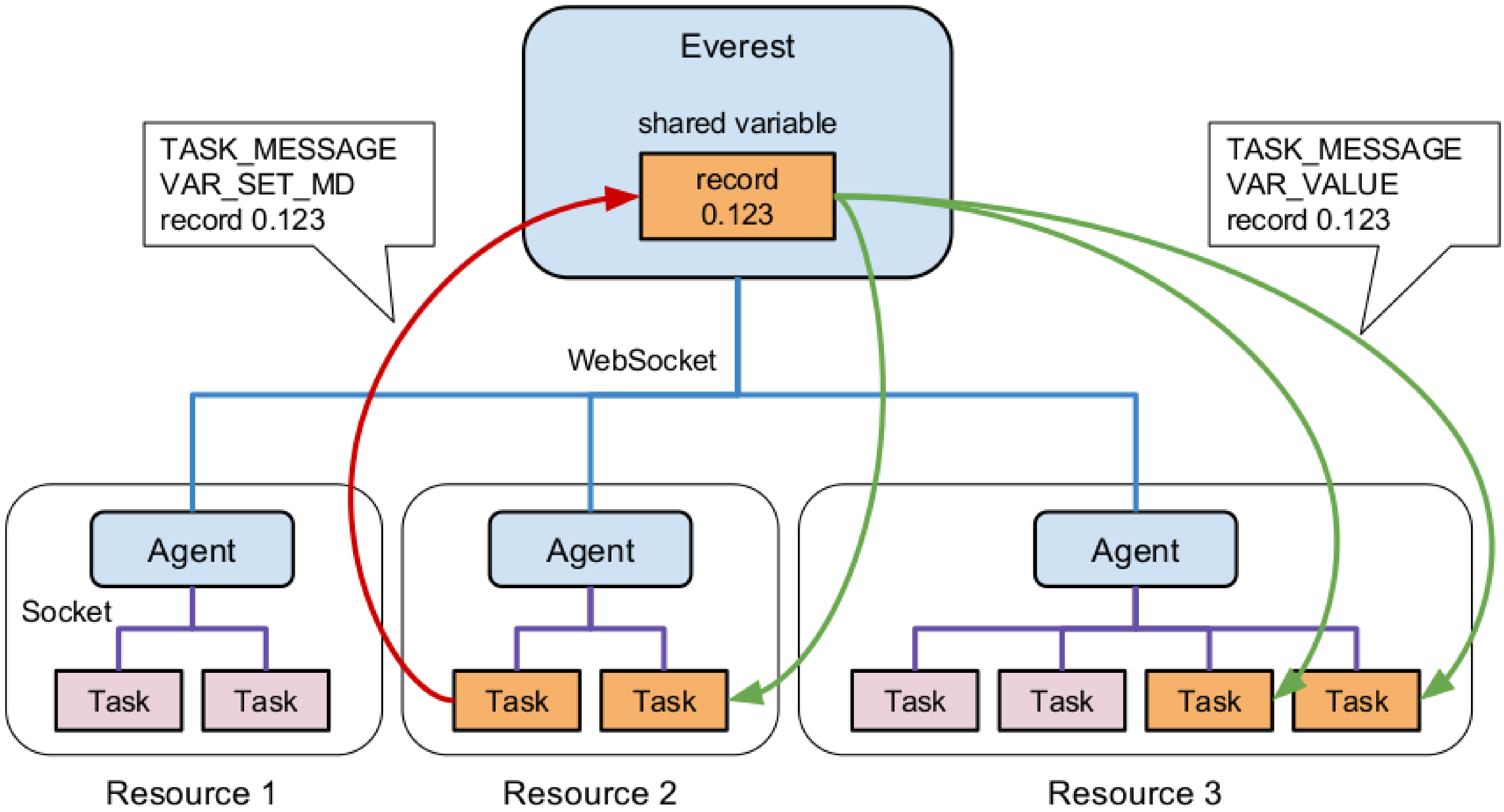
Everest web-based platform, everest.distcomp.org

Describe/Develop/Deploy REST-services representing existing applications



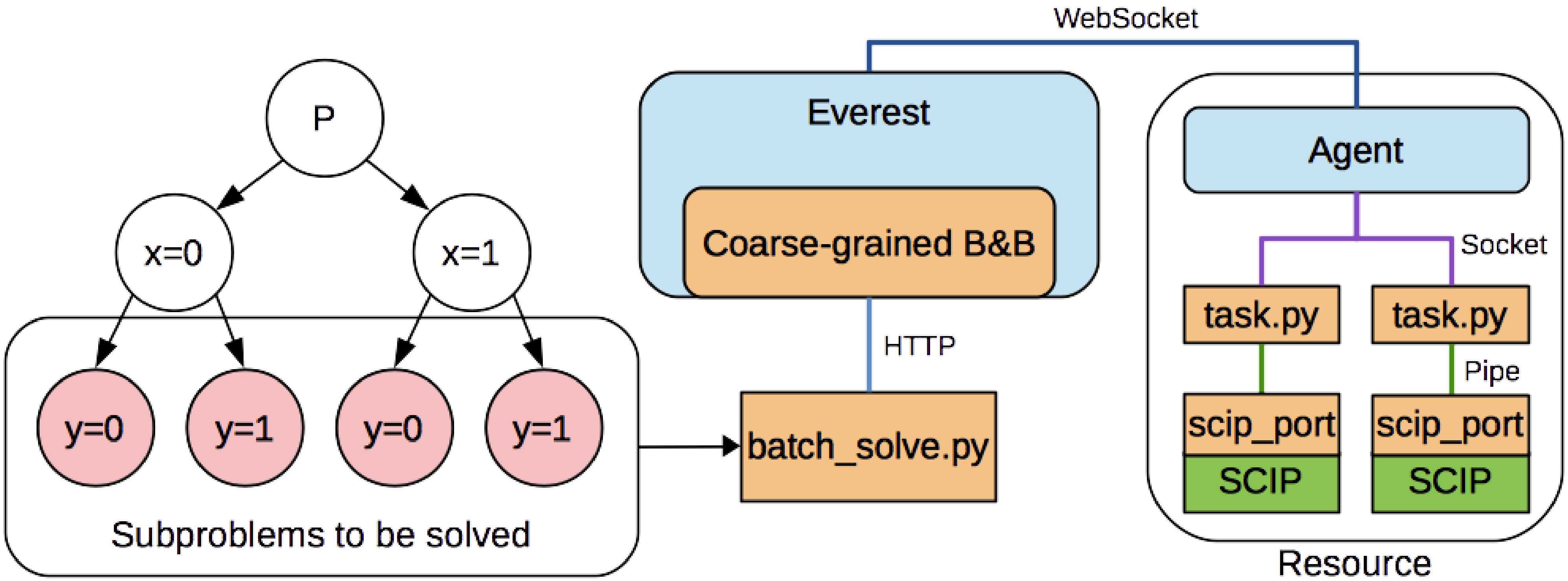
Message exchange via shared variables in Everest

Send message = update shared variable => “multicast” incumbent value



For DDBNB each “task” is running B&B solver, processing MILP/MINLP subproblem

Interaction with solvers



[scip_port], [cbc_port] – solvers' adapters (SCIP, CBC, ...)

[task.py] – simple bi-directional connector between solver and Everest communication mechanism

[DDBNB] – inherits Everest Parameter Sweep generic application, **http://everest.distcomp.org/docs/ps/**

batch_solve.py – Python script to prepare data for [DDBNB].

General scheme of usage

1) Important but informal phase: examine the given Mixed-Integer problem. Choose & test different decomposition scheme and prepare AMPL-stubs *.nl for each subproblem. Use AMPL or Pyomo.

2) Send pack of subproblems to Everest-application [DDB&B] by batch_solve.py

```
./batch_solve.py -s scip -p display/freq=1000 -o tsp70_5  ../TSP_*.nl,
```

or simply use [DDBNB]-Everest application web form

3) Wait for Job completion

Traveling Salesmen Problem Domain Decomposition experiment

$$V \doteq 1{:}N;\ \mathcal{A} \doteq \left\{(i,j) \in V \times V : i \neq j\right\};\ \mathcal{E} \doteq \left\{(i,j) \in V \times V : i > j\right\}$$

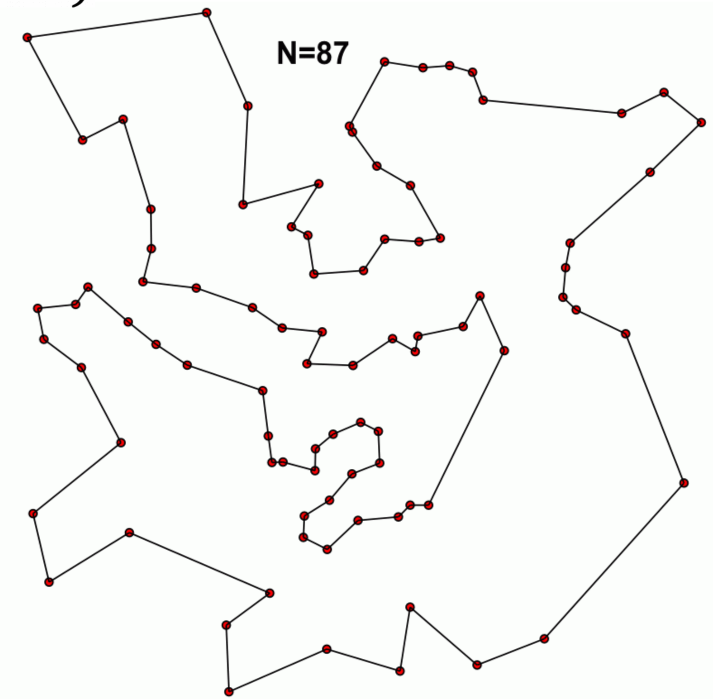
$$\sum_{(i,j)\in\mathcal{E}} d_{ij} x_{ij} \to \min_{x_{ij}, f_{ij}} \quad s.t.:$$

$$\sum_{j\in V, i>j} x_{ij} + \sum_{j \in V, i<j} x_{ji} = 2, \quad i \in V;$$

$$f_{ij} \le \begin{cases} N, & \text{if } i = 1 \\ N-1, & \text{if } i > 1 \end{cases} \cdot \begin{cases} x_{ij}, & i < j \\ x_{ji}, & i > j \end{cases}, (i,j) \in \mathcal{A};$$

$$\sum_{j:(i,j)\in\mathcal{A}} f_{ij} - \sum_{j:(i,j)\in\mathcal{A}} f_{ji} \le \begin{cases} N-1, & i = 1 \\ -1, & i > 1 \end{cases}, \quad i \in V;$$

$$\sum_{j:(i,j)\in\mathcal{A}} f_{ij} \ge 1,\ i \in V;\ x_{ij} = \{0,1\}.$$



“Random” selection of x_{ij} to decompose doesn’t give speed-up

Heuristic rule: sort $\{d_{ij}\}$ in ascending order, and decompose by a few of $x_{i'j'}$:=0|1 from the beginning of the sorted $\{d_{i'j'}\}$ (may be to get subproblems, “balanced” by incumbents ??)

Subproblems has been generated as AMPL-stubs by AMPL script

Everest resources (used for experiments below)

Everest | Applications | Jobs | Resources | Groups | Documentation | About | vladimirv

Resources

Update | New resource

Filter by state: ANY | ONLINE | OFFLINE | My resources

Name	Type	Total Slots	Free Slots	Max Tasks	Total Tasks	Running Tasks	Agent Version	Owner
fujiRestOpt	local	4 ✓	4	4	0	0	2.0	vladimirv
irbis1	local	8 ✓	8	8	0	0	2.0	vladimirv
irbis1_light	local	24	24	24	0	0	2.0a1	vladimirv
mvs10p	slurm	4320	0	400	0	0	2.0	ssmir
restopt-vm1	local	8 ✓	8	8	0	0	2.0	vladimirv
restopt-vm2	local	8 ✓	8	8	0	0	2.0	vladimirv
symbol	local	1	1	1	0	0	2.0	polunovskiy
test	docker	12	12	12	0	0	2.1	sol
ui4.kiae	slurm	12192	0	256	0	0	2.0	ssmir
vvvolx	local	4 ✓	4	4	0	0	2.0	vladimirv

First | Previous | 1 | Next | Last

Showing 1 to 17 of 17 resources

Running DDBNB by batch_solve

```
[user@host]$
ddbnb/everest/batch_solve.py \
 -p display/freq=1000 -o tsp70_5_4 \
 tsp/TSP_Uniform_70_0*.nl
Job submitted: 5928426e320000b3726a7db1
Job 5928426e320000b3726a7db1 state:READY
Job 5928426e320000b3726a7db1 state:RUNNING
Job 5928426e320000b3726a7db1 state:DONE
Downloading job's log...
Best solution 10.034970 (optimal) for
1/output/stub1.sol saved to tsp70_5_4.sol

[user@host]$ ls
tsp70_5_4.log  tsp70_5_4.sol
```

Job Info | Inputs | Outputs | Share | Tasks

Plan File	tsp70_5_4.plan
Stubs & options	tsp70_5_4.zip

Job Info | Inputs | Outputs | Share | Tasks

Application	Coarse-grained B&B
State	RUNNING
Submitted	26 May 2017 22:27:35
Finished	
Info	RUNNING: 24 / WAITING: 8 / DONE: 0 / FAILED: 0 / CANCELLED: 0 / Estimated time left: unknown
Log	view

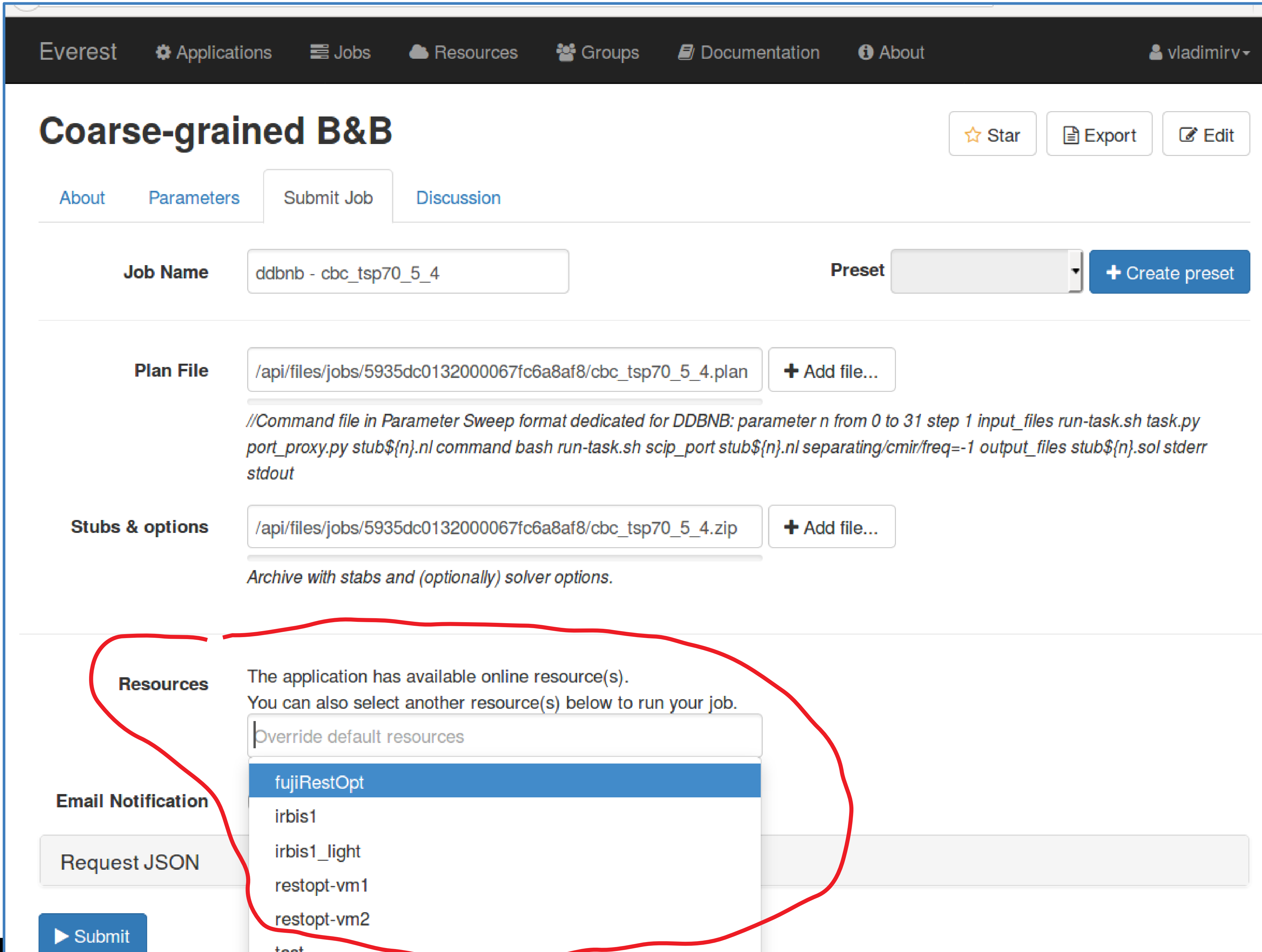
Job Info | Inputs | Outputs | Share | Tasks

Application	Coarse-grained B&B
State	DONE
Submitted	26 May 2017 22:27:35
Finished	26 May 2017 22:28:28
Info	RUNNING: 0 / WAITING: 0 / DONE: 32 / FAILED: 0 / CANCELLED: 0
Log	view

Job Info | Inputs | Outputs | Share | Tasks

Results	results.zip

DDBnB WebUI (Submit Job Form)



DDB&B WebUI (View states of tasks)

Everest | Applications | Jobs | Resources | Groups | Documentation | About | vladimirv

ddbnb - cbc_tsp70_5_4

Job Info | Inputs | Outputs | Share | Tasks

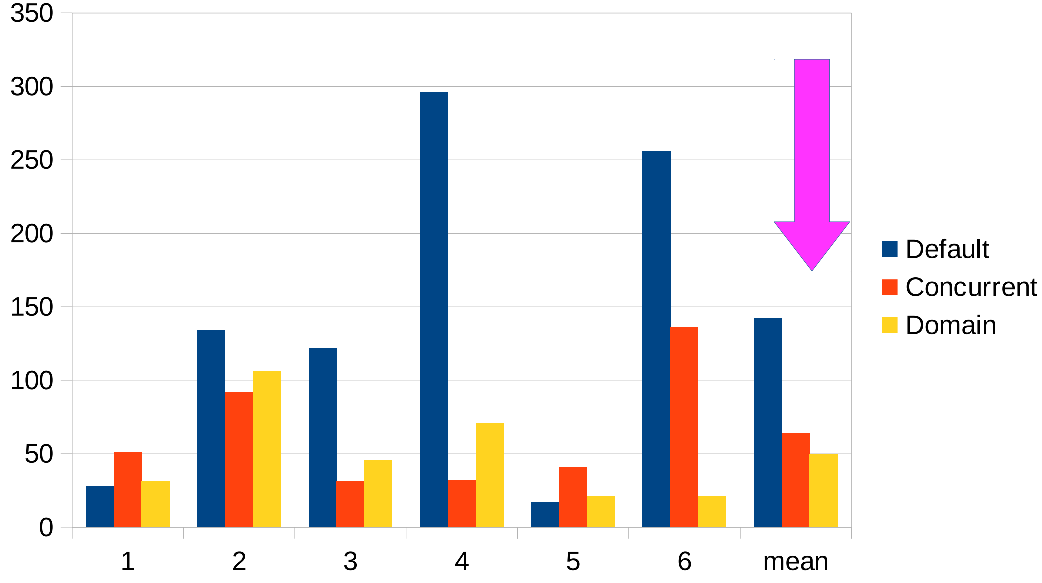
Task Id	State	Resource	Created	Stage In	Wait Time	Runtime	Stage Out	Finished	Files
0	RUNNING	restopt-vm1	06 Jun 2017 01:32:34	0.00 KB	0s				
1	DONE	vvvolx	06 Jun 2017 01:32:34	0.00 KB	0s	4m45s	33.25 KB	06 Jun 2017 01:37:20	
2	DONE	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s	6m0s	33.42 KB	06 Jun 2017 01:38:35	
3	RUNNING	restopt-vm1	06 Jun 2017 01:32:34	0.00 KB	0s				
4	RUNNING	fujiRestOpt	06 Jun 2017 01:32:34	0.00 KB	0s				
5	DONE	vvvolx	06 Jun 2017 01:32:34	0.00 KB	0s	4m45s	33.37 KB	06 Jun 2017 01:37:20	
6	DONE	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s	6m0s	33.42 KB	06 Jun 2017 01:38:35	
7	DONE	fujiRestOpt	06 Jun 2017 01:32:34	0.00 KB	0s	6m25s	33.31 KB	06 Jun 2017 01:39:01	
8	DONE	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s	6m0s	33.43 KB	06 Jun 2017 01:38:35	
9	DONE	vvvolx	06 Jun 2017 01:32:34	0.00 KB	0s	5m5s	33.02 KB	06 Jun 2017 01:37:40	
10	DONE	restopt-vm2	06 Jun 2017 01:32:34	0.00 KB	0s	5m40s	33.38 KB	06 Jun 2017 01:38:15	
11	RUNNING	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s				
12	DONE	restopt-vm2	06 Jun 2017 01:32:34	0.00 KB	0s	5m50s	33.31 KB	06 Jun 2017 01:38:25	
13	RUNNING	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s				
14	RUNNING	vvvolx	06 Jun 2017 01:32:34	0.00 KB	0s				
15	DONE	fujiRestOpt	06 Jun 2017 01:32:34	0.00 KB	0s	6m25s	33.30 KB	06 Jun 2017 01:39:00	
16	DONE	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s	6m0s	33.44 KB	06 Jun 2017 01:38:35	
17	RUNNING	irbis1	06 Jun 2017 01:32:34	0.00 KB	0s				
18	DONE	restopt-vm1	06 Jun 2017 01:32:34	0.00 KB	0s	6m25s	33.38 KB	06 Jun 2017 01:39:00	
19	DONE	vvvolx	06 Jun 2017 01:32:34	0.00 KB	0s	4m40s	33.72 KB	06 Jun 2017 01:37:15	

First | Previous | 1 | 2 | Next | Last

There are 30 processors from 5 hosts dedicated to the application

Experiments: Domain Decomposition vs Concurrent

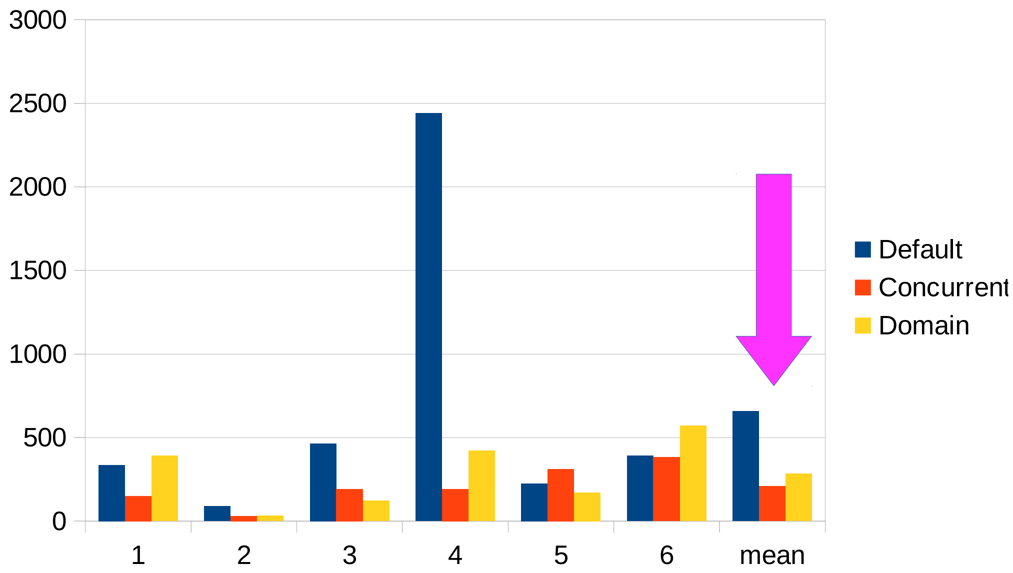
N = 70



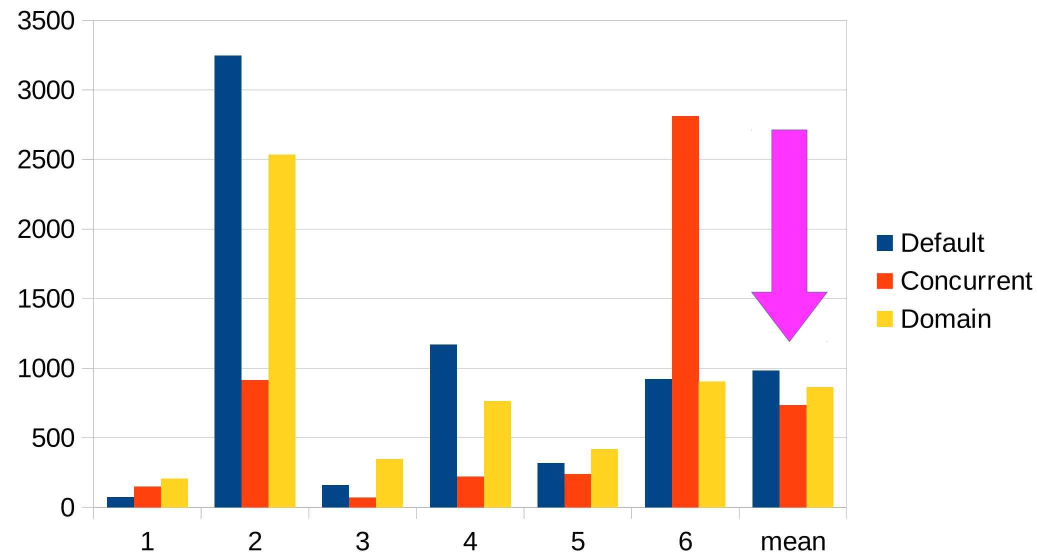
Run times for several TSP instances:
N – number of cities
6 random instances for every N

7 different settings for concurrent case
… MORE subproblems for DD!!!

N = 80



N = 90



Experiments, Domain Decomposition & Concurrent

batch_solve.py -pf p1.txt p2.txt … p7.txt ... -o tsp70_3 ../70_3/TSP_70_*.nl

p#.txt – files with SCIP parameters (*nodeselection/childsel* = {d,u,p,i,…})

tsp##_(1:6) – six TSP problem randomly generated for a given N

0 – only Concurrent (7 p#.txt and one stub, 7 problems solving in parallel)

1 – DD&C (DD by ONE binary variable, 2 *.nl files x 7 p#.txt = 14 problems)

2 – DD&C (DD by TWO binary variable, 4 *.nl x 7 p#.txt = 28 problems)

3 – DD&C (DD by 3 binary variables, 8 *.nl x 7 p#.txt = 56 problems)

4 – DD&C (DD by 4 binary variables, 16 *.nl x 7 p#.txt =112 problems)

Heterogeneous environment (2xVM/8cores + 2xCS/4cores + CS/8cores = 32 cores)

In tables below elapsed time in seconds.

N=70	SCIP	0	1	2	3	4
tsp70_1	47	51	43	42	48	98
tsp70_2	92	66	97	42	67	135
tsp70_3	142	74	82	122	67	122
tsp70_4	32	31	31	52	52	110
tsp70_5	57	42	36	47	68	100
tsp70_6	47	37	43	57	63	117
sum	**417**	**301**	**332**	**362**	**365**	**682**

N=80	SCIP	0	1	2	3	4
tsp80_1	197	122	72	67	122	218
tsp80_2	21	26	37	53	53	141
tsp80_3	242	149	97	118	118	170
tsp80_4	302	57	48	63	149	157
tsp80_5	107	84	58	77	74	165
tsp80_6	707	276	186	132	139	405
sum	**1576**	**714**	**498**	**510**	**655**	**1256**

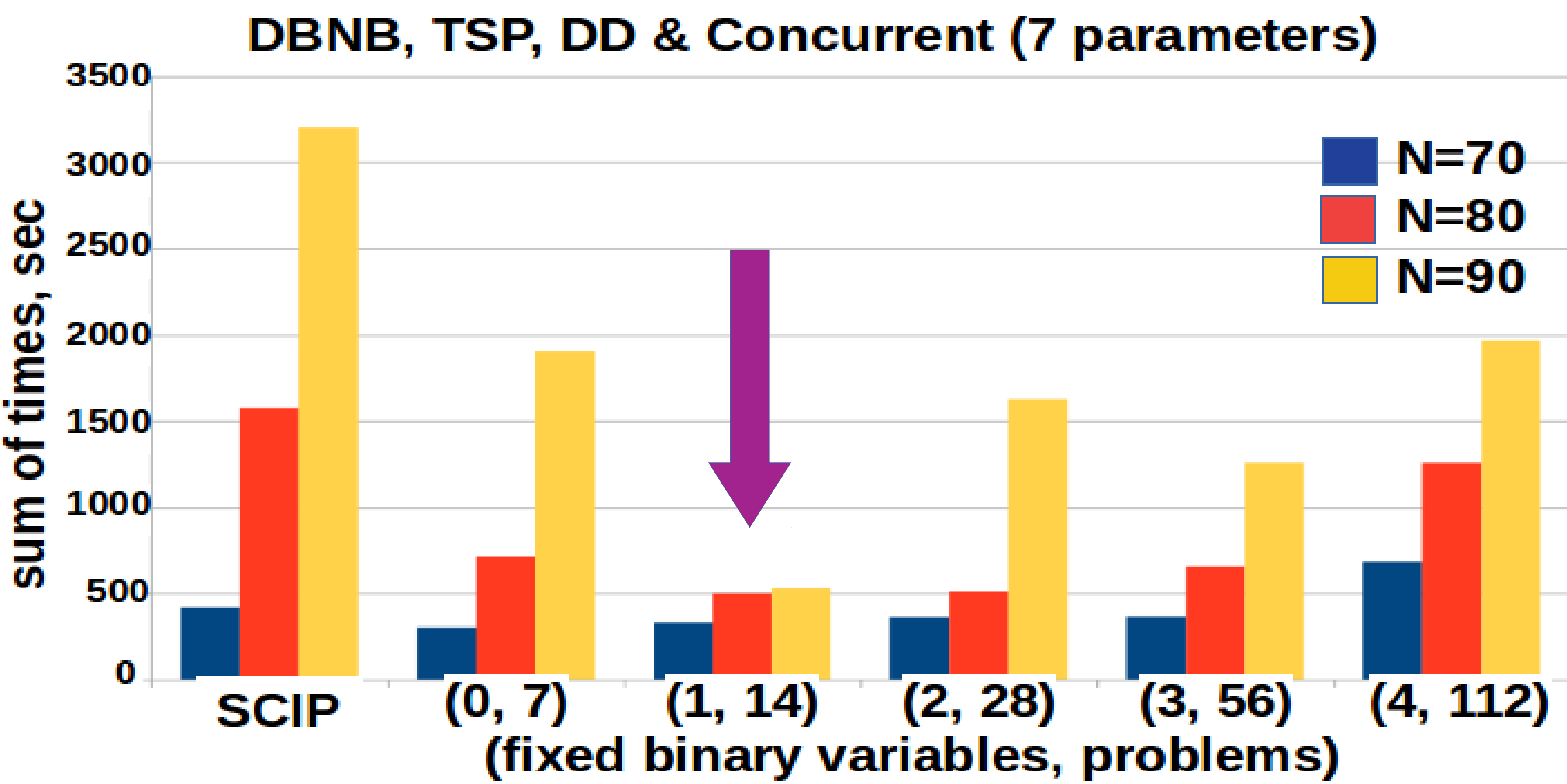
Experiments, Domain Decomposition & Concurrent (2)

For used computing environment:

2xVM/8cores + 2xCS/4cores + CS/8cores = 32 cores

In average, the best became: DD into 2 subproblems and solving in parallel by 7 different search traversal options

N=90	SCIP	0	1	2	3	4
tsp90_1	153	114	77	132	185	221
tsp90_2	646	559	82	167	174	709
tsp90_3	202	57	47	62	79	173
tsp90_4	372	87	132	267	233	328
tsp90_5	341	179	67	117	118	230
tsp90_6	1488	907	124	882	468	303
sum	**3202**	**1903**	**529**	**1627**	**1257**	**1964**



Subtotal & Next topic DDBNB for Global Optimization

1) Смирнов С.А., Волошинов В.В. *Эффективное применение пакетов дискретной оптимизации в облачной инфраструктуре на основе эвристической декомпозиции исходной задачи в системе оптимизационного моделирования AMPL* // Программные системы: теория и приложения, No 28, 2016, с. 29–46
2) Волошинов В.В., Смирнов С.А.. *Оценка производительности крупноблочного алгоритма метода ветвей и границ в вычислительной среде Everest* // Программные системы: теория и приложения, т. 8, № 1, 2017, 105–119
3) Voloshinov V., Smirnov S. and Sukhoroslov, O., *Implementation and Use of Coarse-grained Parallel Branch-and-bound in Everest Distributed Environment* // Procedia Computer Science, 108, 2017, pp. 1532-1541
4) Smirnov S., Voloshinov V. *Implementation of Concurrent Parallelization of Branch-and-bound algorithm in Everest Distributed Environment* // Procedia Computer Science, 119, 2017, pp. 83–89

What else? Global optimization problems with examples of combinatorial geometry problems.

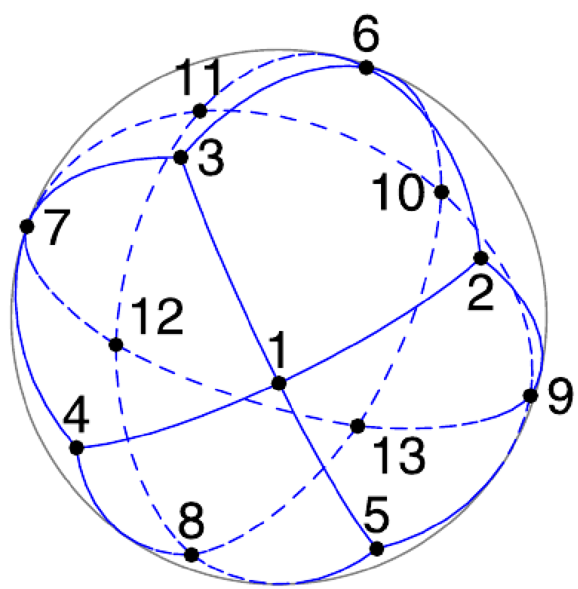
Tammes problem (optimal packing of circles on a sphere): arrange N points on a unit sphere to maximize minimal pairwise distance

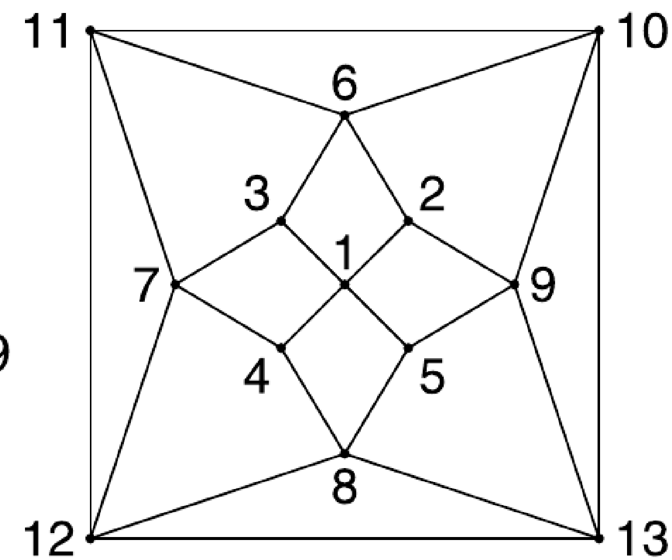
$$\min_{1\leqslant i<j\leqslant N}\{\|x_i-x_j\|\}\to\max_{x_i\in\mathbb{R}^3,\ i=1:N}\quad s.t.:\ \|x_i\|=1\ (i=1:N).$$

Oleg Musin, Alexey Tarasov: (**2012**) *The strong* ***thirteen spheres problem***. Discrete & Computational Geometry, 48(1):128–141; (**2015**) *The Tammes Problem for **N=14**, Experimental Mathematics*, 24:4, 460-468

It is not difficult to make a conjecture about optimal arrangement, but it is very difficult to prove that is global optimum!

The optimal configuration of 14 points was conjectured more than 60 years ago, but **computer-assisted proof** has been done on 2015 **by enumeration of the irreducible contact graphs.**





Tammes problem (2)

Formulated as global optimization with bilinear functions

$$z \to \min_{x_i \in \mathbb{R}^3,\ i=1:N},$$

$$x_i^{\mathsf{T}} x_j = \sum_{k=1:3} x_{i,k} x_{j,k} \leqslant z, \quad (1 \leqslant i < j \leqslant N)$$

$$\|x_i\|^2 = \sum_{k=1:3} (x_{i,k})^2 = 1, \quad (i=1:N)$$

$$x_{1,1} = x_{1,2} = 0, x_{1,3} = 1,$$
$$x_{2,1} = 0, x_{2,1} \geqslant 0,$$
$$x_{i+1,3} \leqslant x_{i,3} \ (1 \leqslant i \leqslant N-1)$$

Auxiliary, antisymmetric, constraints:
- set **1st** point equal to a “pole” (0,0,1);
- **2nd** point lies in ZOY plane, “positively”
- “anti-renumeration”, **3d** coordinate in ascending order.

Non-convex problem; no integer variables, many local optimums.

SCIP solver supports non-convex problems with polynomial functions in constraints.

For problems with polynomials SCIP consumes a lot of memory

```
*******    For Tammes, N=8 ************
 time | node   | left   |LP iter|LP it/n| mem |...|cuts |confs|strbr|  dualbound    | primalbound  |  gap
 2737s|  7203k|   2683k| 75504k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362514e-01 | 2.612038e-01 |  91.71%
 2738s|  7204k|   2683k| 75508k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362564e-01 | 2.612038e-01 |  91.70%
 2738s|  7204k|   2683k| 75512k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362635e-01 | 2.612038e-01 |  91.69%
 2738s|  7205k|   2683k| 75517k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362680e-01 | 2.612038e-01 |  91.68%
 2738s|  7205k|   2683k| 75521k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362759e-01 | 2.612038e-01 |  91.67%
 2738s|  7206k|   2683k| 75526k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362803e-01 | 2.612038e-01 |  91.67%
 2739s|  7206k|   2684k| 75530k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362885e-01 | 2.612038e-01 |  91.66%
 2739s|  7207k|   2684k| 75535k|   10.5 |  29G|...|  49M|    0 |    0 | 1.362926e-01 | 2.612038e-01 |  91.65%

SCIP Status        : solving was interrupted [memory limit reached]
Solving Time (sec) : 2739.08
Solving Nodes      : 7207031
Primal Bound       : +2.61203825123857e-01 (4 solutions)
Dual Bound         : +1.36292622736940e-01
Gap                : 91.65 %
```

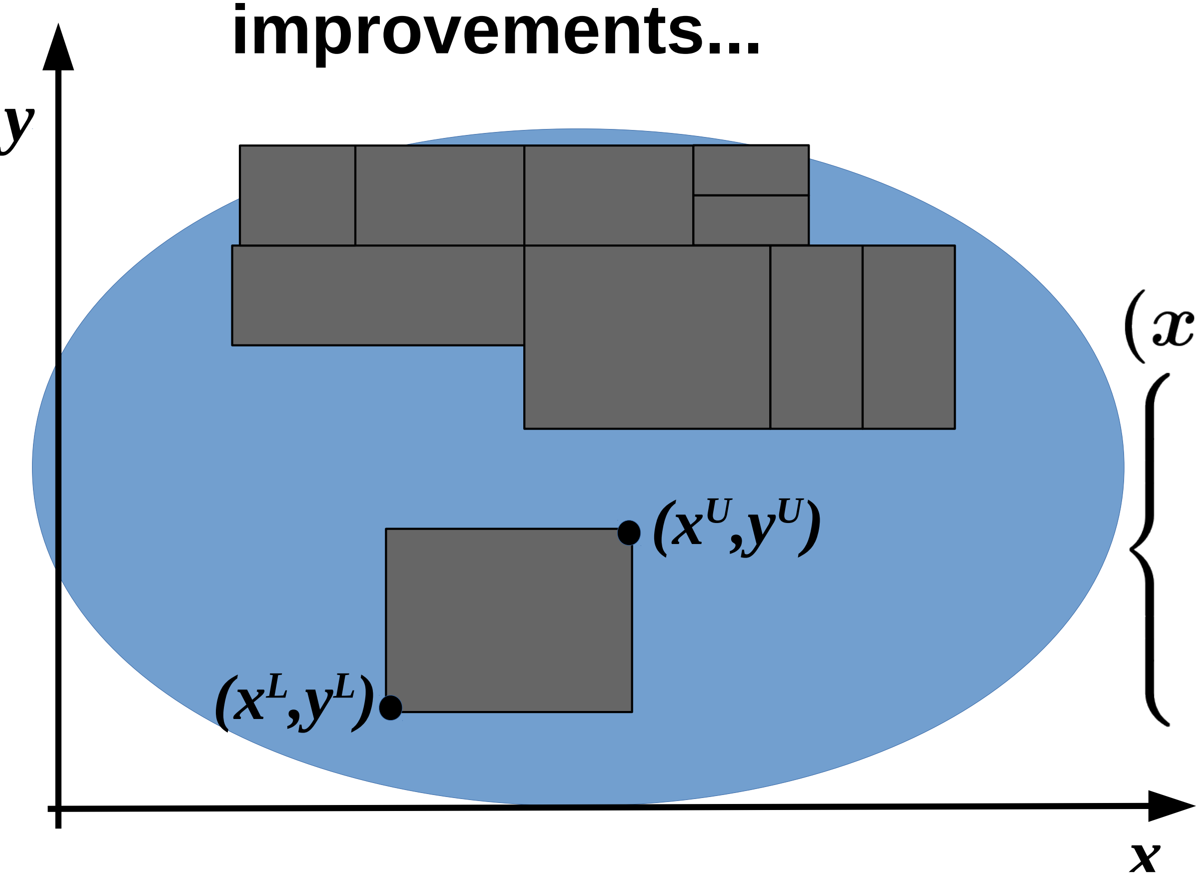
Excerpt of SCIP log when solving Tammes problem with N=8.

Explanation and workaround – further.

Convex relaxation of bilinear functions (for BnB)

Convex relaxation to get lower bound on “hyper-rectangles”

Garth P. McCormik envelopes (1976) and their improvements...



$$(x, y) \in [x^L, x^U] \times [y^L, y^U]:$$

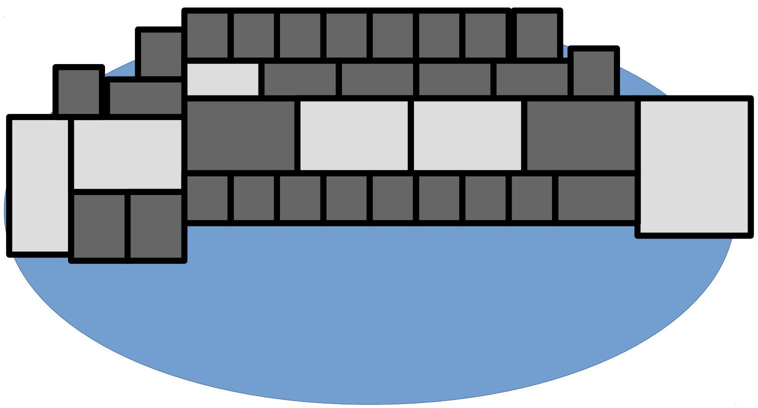
$$\begin{cases} xy \geqslant x^L y + y^L x - x^L y^L \\ xy \geqslant x^U y + y^U x - x^U y^U \\ xy \leqslant x^L y + y^U x - x^L y^U \\ xy \leqslant x^U y + y^L x - x^U y^L \end{cases}$$

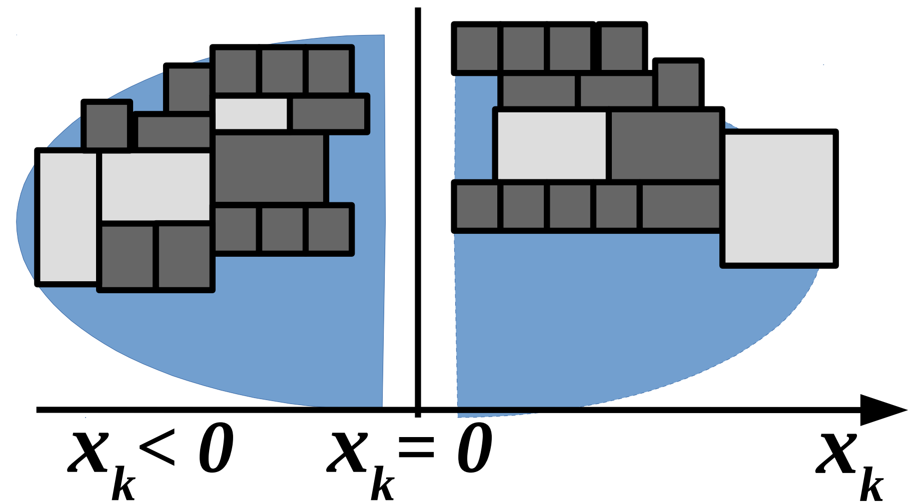
The more rectangles (smaller ones!), the tighter approximation we have, the less the BnB gap!

The more rectangles – the more memory to store them!

Why domain decomposition may help?

Reducing the volume of sub-domain by half gives "twice" less memory for storing BnB search tree (rectangle partitions covering sub-domain)





$$([p_k = \pm 1, k = 2{:}K])$$

$$z \to \min_{x_i \in \mathbb{R}^3,\ i = 1:N},$$

$$\dots$$

$$p_k \cdot x_{k,1} \leqslant 0, (k = 2{:}K)$$

- **DD parameters, 2^{K-1} sub-problems**
- **the same objective function**
- **the same constraints**
- **additional DD constraints**

This simple DD has drawbacks: 1) generic limit on number of sub-problems ($\leqslant 2^{N-1}$); 2) DD is “unbalanced” regarding geometric sense, e.g. $\{p_k=1, k=2,K\} \Rightarrow x_{k,1} \leqslant 0$ $(k=1{:}K)$ – obviously “not optimal”

"Advanced" Domain Decomposition strategy

Take $K<N$ and subset of distinct indices $\mathcal{N}=\{n_k: k=1:K,\ 2\leqslant n_k\leqslant N\}$.

Then for some $M\leqslant K$ take set ${}^K S_M$ of all choices of M indices from K-subset of $\mathcal{N}$. So, $s\in {}^K S_M \Rightarrow s=(k_1, k_2,\dots, k_M)$ and

$\left|{}^K \mathrm{S}_M\right| = \binom{K}{M} := \frac{K!}{M!(K-M)!}$ **which may be much more than K.**

$$\{p_s=\pm 1, s\in {}^K \mathrm{S}_M\}$$

$$z \to \min_{x_i\in\mathbb{R}^3,\ i=1:N},$$

$$\dots$$

$$p_s\cdot\sum_{m=1}^{M} x_{n_{k_m},1} \leqslant 0, \left(s\in {}^K \mathrm{S}_M\right)$$

- **DD parameters, $2^{\wedge}|{}^K S_M|$ sub-problems**
- **the same objective function**
- **the same constraints**

- additional DD constraints

This rule gives much more subdomains which might become more "balanced" than those given by previous "simple" strategy.

Cardinality of $|{}^K S_M|$ increases dramatically even for not very large K and M. But it may be restrict yourself with some subset $S' \subset {}^K S_M$

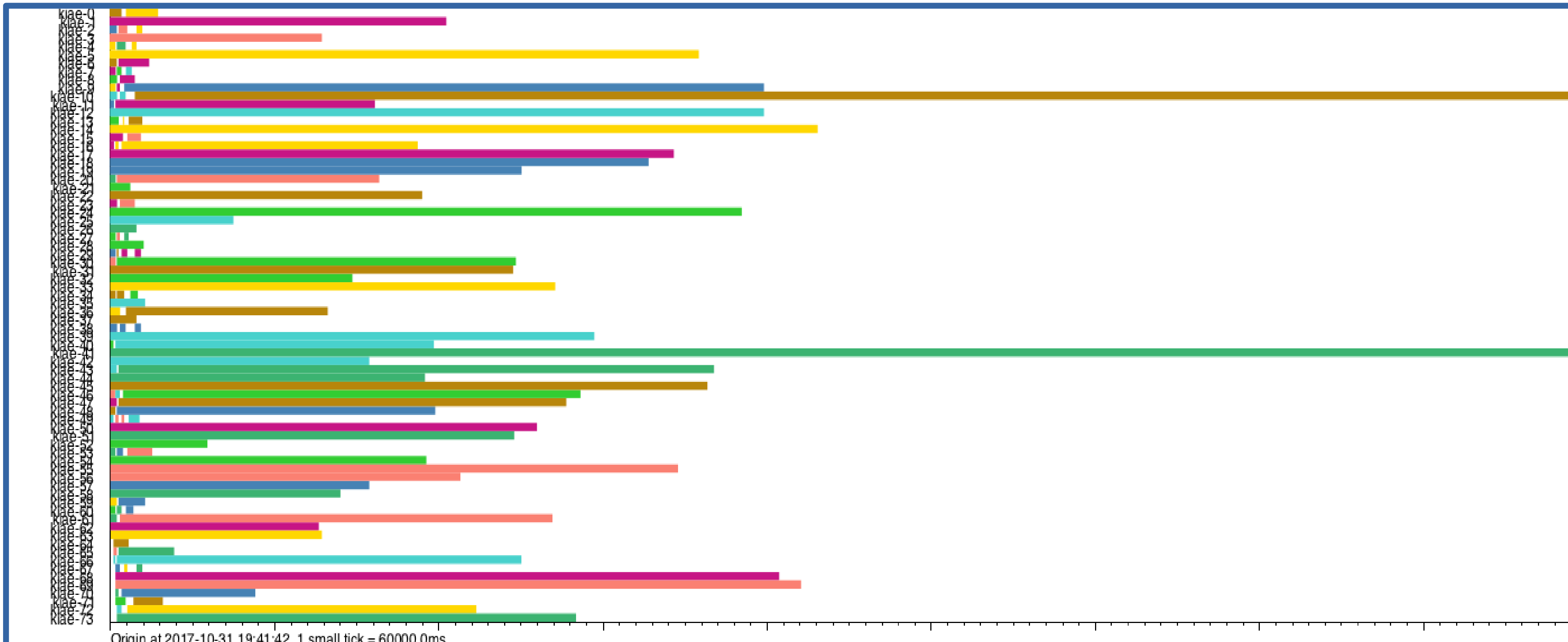
Computing experiments with Tammes N=8

N=8. It could not be solved by one SCIP.

Simple DD rule, K=7, 128 subproblems,

Cluster HPC4, NRC "Kurchatov Institute" (4th in Russian Top50)

DDBNB solved the problem in 100 minutes.



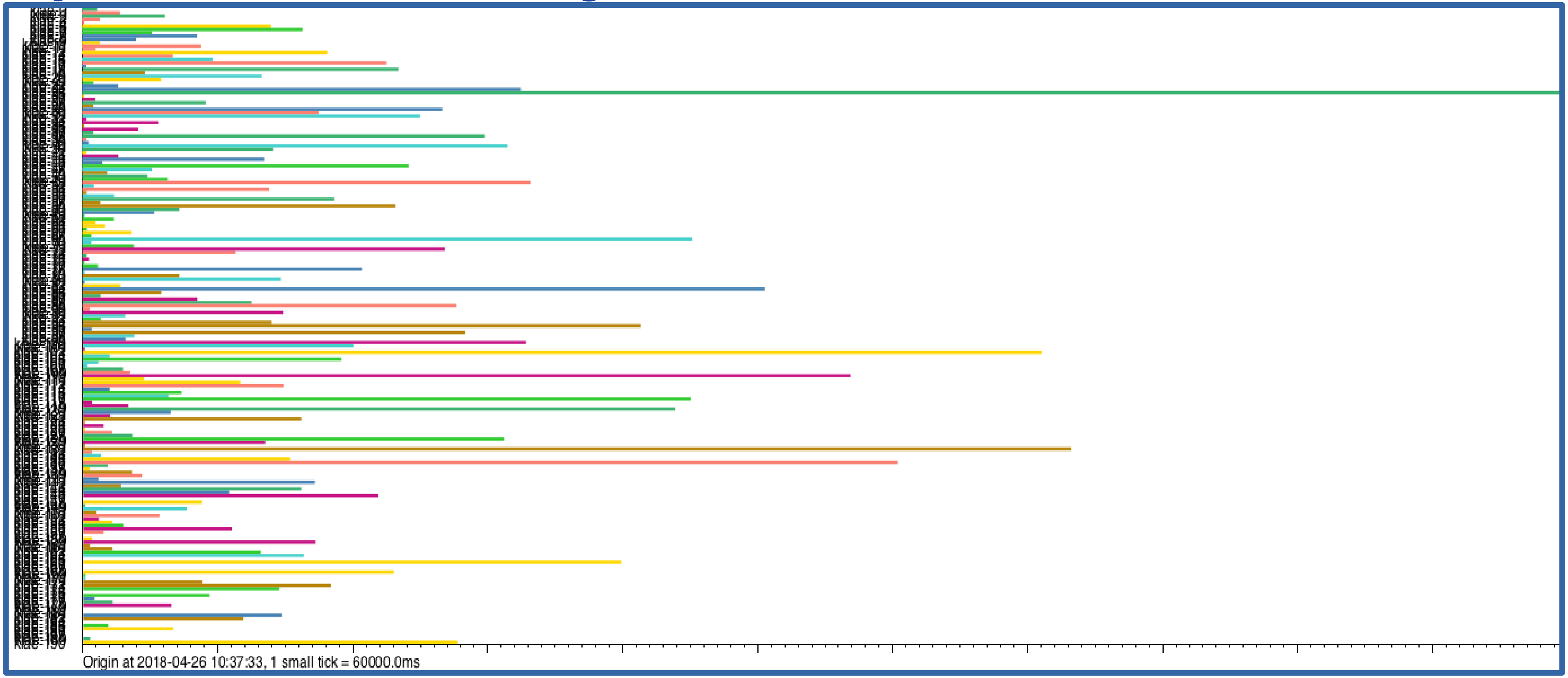
Has been solved, but subproblems are very unbalanced…

Memory limit = 16Gb per CPU.

Computing experiments with Tammes N=9

Simple DD rule, K=8, 256 subproblems, “Kurchatov” cluster. DDBNB solved in ~70 hours (3 days)?

Advanced DD rule, K=8, $\mathcal{N}=\{2,3,4,5,6,7,8,9\}$, M=7, 256 subproblems, “Kurchatov” cluster. DDBNB solved in 120 minutes (160 min 2nd run)! But we see unbalancing still. No dynamic load balancing...



Thomson's problem

Thomson problem: minimize total Coulomb energy of N unit charges on a sphere $\sum_{1\leqslant i<j\leqslant N} \|x_i - x_j\|^{-1} \to \min_{x_i\in\mathbb{R}^3,\ i=1:N}$ s.t.: $\|x_i\| = 1\ \ (i=1:N)$

$$z_{ij} \doteq \|x_i - x_j\|^{-1}$$

$$\sum_{i,j=1,i<j}^{N} z_{ij} \to \min_{x_i, z_{ij}} \quad s.t.:$$

$$z_{ij}^2 \,(x_i - x_j)^{\mathsf{T}}(x_i - x_j) = 1\ \ (1\leqslant i<j\leqslant N);$$

$$x_i^{\mathsf{T}} x_i = 1\ \ (i=1:N),\ x_i \in \mathbb{R}^3, z_{ij}\in\mathbb{R}.$$

$$x_{1,1}=x_{1,2}=0, x_{1,3}=1,$$
$$x_{2,1}=0, x_{2,1}\geqslant 0,$$
$$x_{i+1,3} \leqslant x_{i,3}\ \ (1\leqslant i\leqslant N-1)$$

Auxiliary, antisymmetric, constraints: the same as for Tammes problem

Another “hard-to-prove-global-optimality” problem with centuries of history.

E.g. **the case N=5 has got computer-aided proof in 2013**:

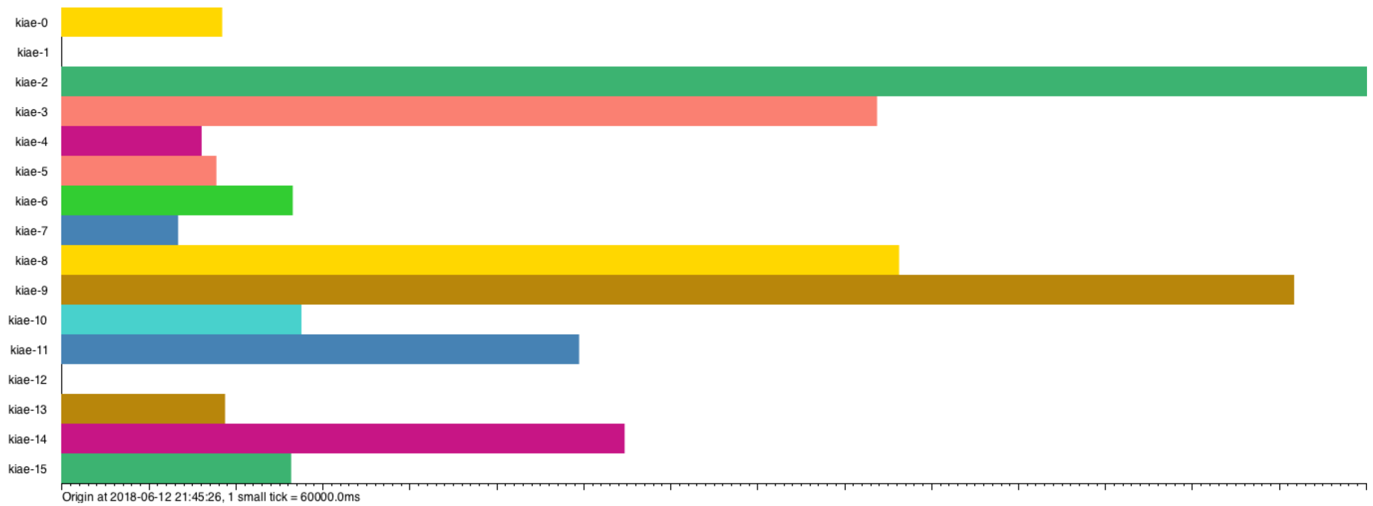
R. Schwartz. The 5-electron case of Thomson's problem // Exp. Math., 22(2):157--186, 2013.

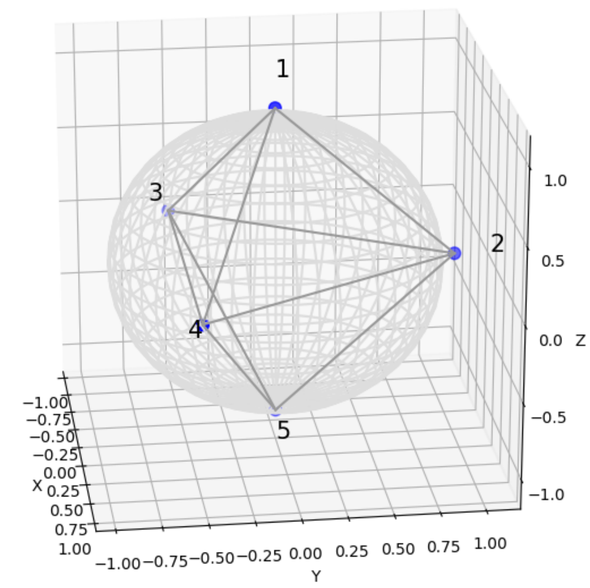
-30 pages of theory and ~30 min of computing on MacBook Pro...

Thomson problem, N=5

One SCIP 5.0.1 solver at one CPU of "Kurchatov" cluster. Solved in ~600 minutes (SCIP 6.0.0 – 560 min).

Advanced DD rule, K=4, $\mathcal{N}$={2,3,4,5}, M=3, 16 subproblems, "Kurchatov" cluster, DDBNB solved in 160 min.





ParaSCIP, "Kurchatov" cluster, 8 cores, 86 min (four-times more effective than DDBNB) with fine-grained parallelization of SCIP-BNB and dynamic load balancing...

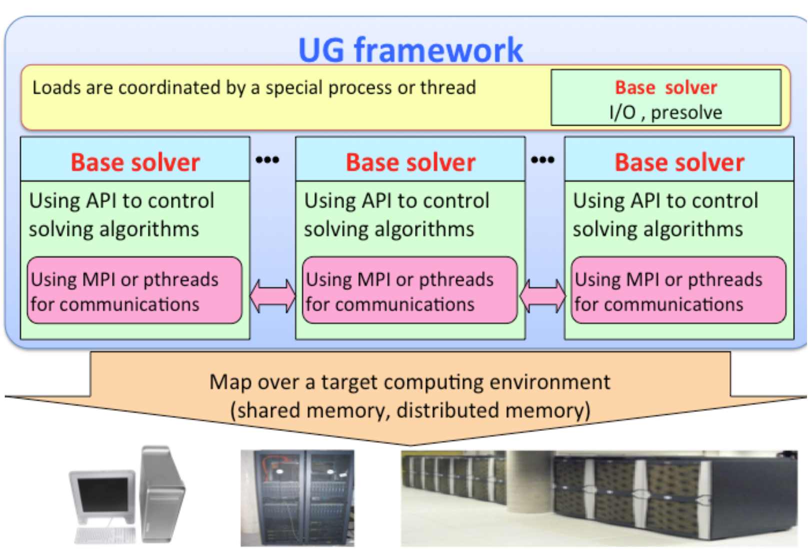
ParaSCIP, Zuse Institute Berlin

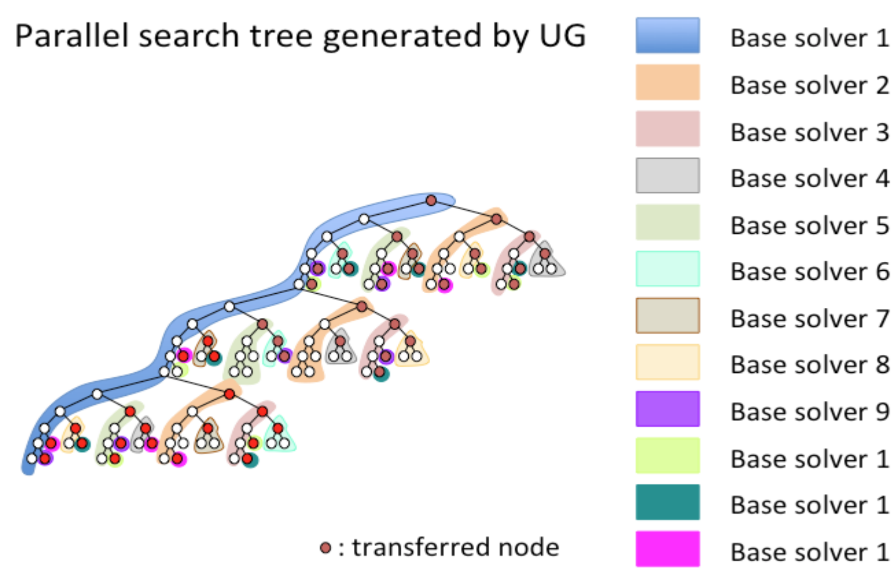
Parallel implementation of B&B via SCIP and MPI for High-Performance Computing environments, http://ug.zib.de/

UG (**Ubiquity Generator**) is a framework to parallelize B&B solvers in a distributed or shared memory computing environment.

ParaSCIP = UG[SCIP, MPI], *FiberSCIP=UG[SCIP,* ***Pthreads****]*, *ParaXpress*=UG[Xpress, MPI],…

Yuji Shinano, Tobias Achterberg, Timo Berthold, Stefan Heinz, Thorsten Koch, *ParaSCIP -- a parallel extension of SCIP*, **2012**





ParaSCIP, Zuse Institute Berlin (2)

Success story of solving open instances from MIPLIB2010 on:

North-German Supercomputing Alliance (Zuse Institute), Germany:

- **HLRN-II**, ~12 000 cores, https://www.hlrn.de/home/view/System2
- **HLRN-III**, ~40 000 cores, SGI Cray, https://-*-/System3

Experiments with 1024 – 12000 cores, 1 – 200 hours

OAK Ridge National Laboratory, USA

- Titan, Cray XK7, ~500000 cores, http://www.olcf.ornl.gov/titan

Experiments with 80 000 cores.

Small experience solving nonlinear problems, MILP - basically

Our input: HPC4/HPC5, NRC "Kurchatov Institute", ~22 000 cores, T–Platforms (458 in world Top500, 4 in Russia Top50)

Experiments with MINLP: Thomson problems (N=5),

Flat Torus Packing problem N=9 – open conjecture has been proved (it took 128 cores * ~16 hours = 2048 CPU*hours)

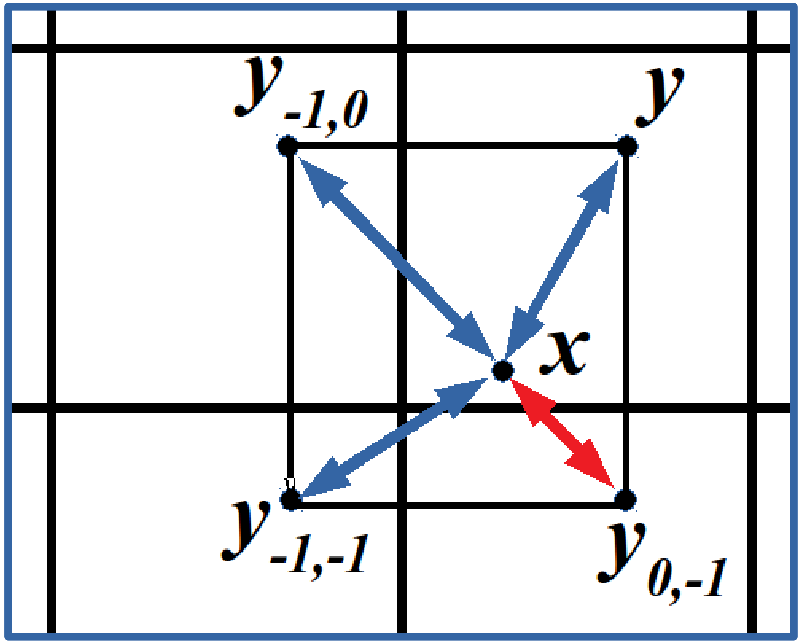
Flat Torus Packing Problem (FTPP), formulation

Metric on Flat Torus ($\mathbb{R}^2/\mathbb{Z}^2$) is induced by Euclidean metric

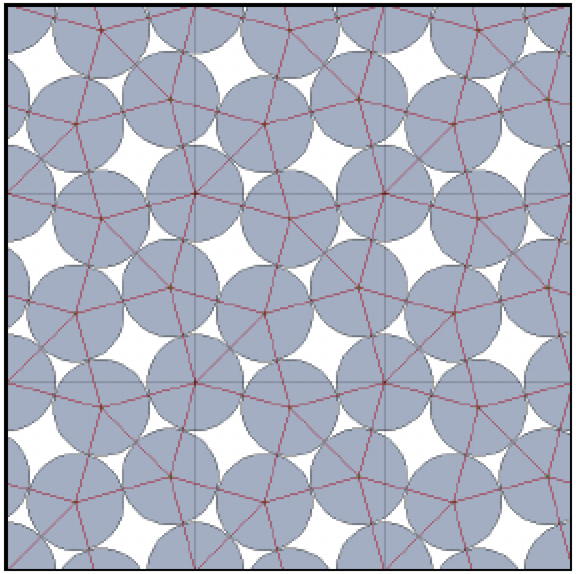
$$x, y \in \mathbb{R}^2,\ d(x,y) \doteq \sum_{k=1:2} \left(\min \{ |x_k - y_k|, 1 - |x_k - y_k| \} \right)^2$$

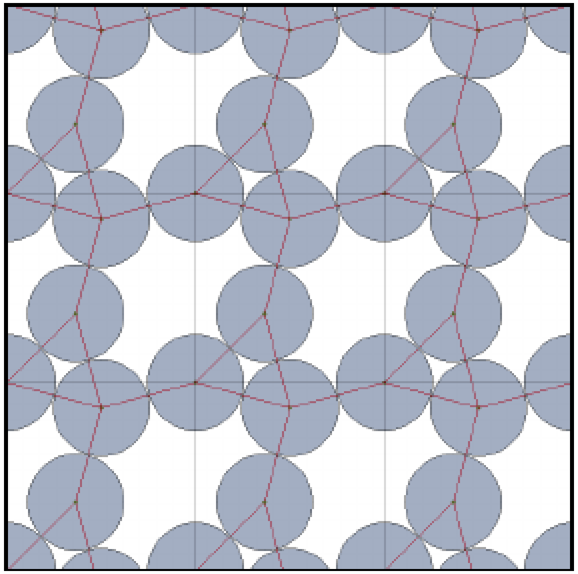
$$\{x_i : i \in 1:N\}, x_i \in [0,1] \otimes [0,1]$$

$$\min_{1 \leqslant i < j \leqslant N} d(x_i, x_j) \to \max_{\{x_i : i \in 1:N\}} : x_i \in [0,1] \otimes [0,1] \quad \textbf{(FTPP)}$$



The problem of "super resolution of images" by space or air survey





Oleg Musin, Anton Nikitenko. Optimal packings of congruent circles on a square flat torus. *Discrete & Computational Geometry*, 55(1):1–20, 2016.

FTPP as NLP with MIxed-Integer variables (e.g. see Dantzig, 1960)

$$i=1:N,\ \ k=1:2,\ \ \mathbf{IJ}\doteq\{(i,j):1\leqslant i<j\leqslant N\},$$

$$y_{ijk}\doteq\min\{|x_{ik}-x_{jk}|,1-|x_{ik}-x_{jk}|\},\ \Big(k=1:2,(i,j)\in\mathbf{IJ}\Big),$$

$$z_{ijk}\doteq-|x_{ik}-x_{jk}|=\min\{x_{jk}-x_{ik},x_{ik}-x_{jk}\}\ \Big(k=1:2,(i,j)\in\mathbf{IJ}\Big),$$

$$\eta_{ijk}\in\{0,1\},\ \zeta_{ijk}\in\{0,1\}\ \Big(k=1:2,\ (i,j)\in\mathbf{IJ}\Big).$$

2N **continuous variables** x_{ik}, y_{ijk}, z_{ijk} **and** ***2N(N-1)*** **binary** η_{ijk}, ζ_{ijk}.

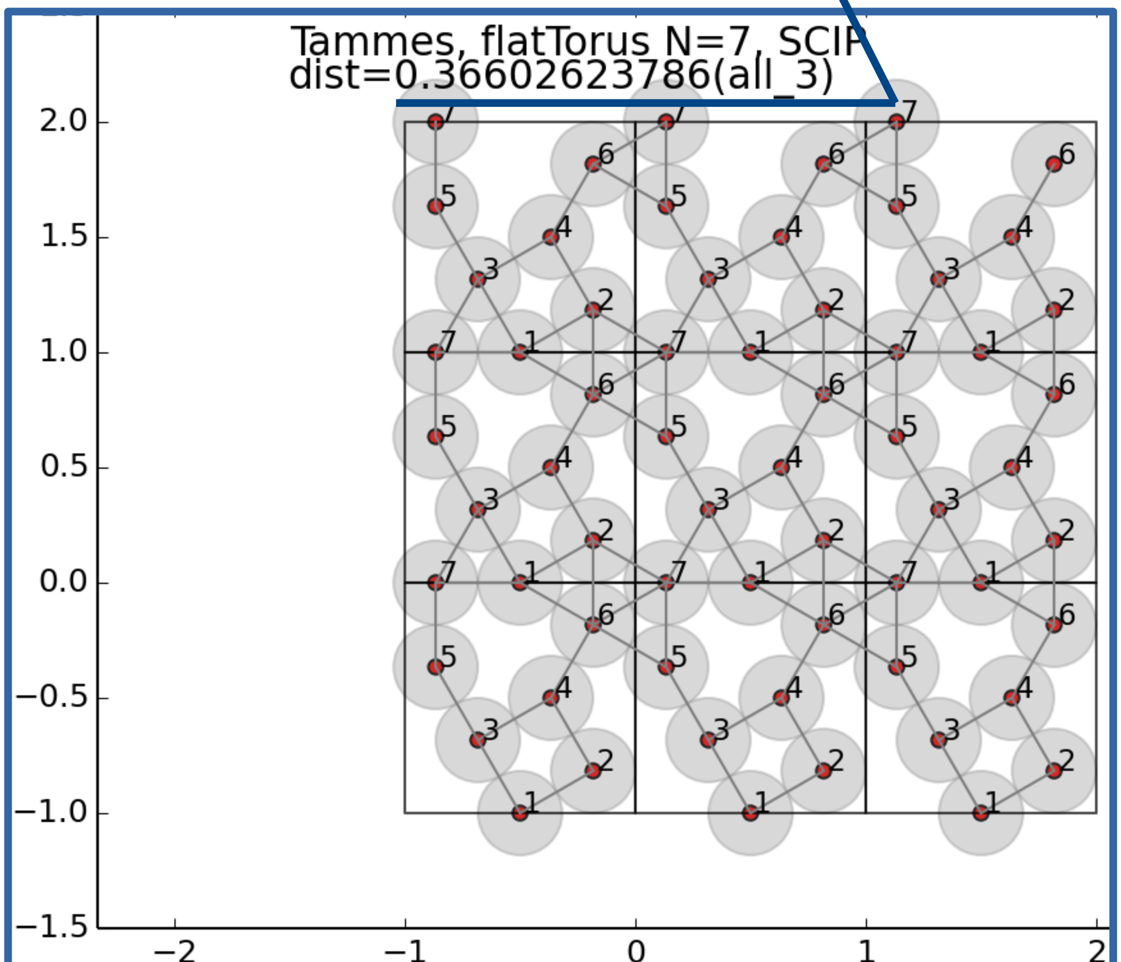
$$\begin{aligned}
&D\to\max(\text{with vars. } x_{ik},y_{ijk},z_{ijk},\eta_{ijk},\zeta_{ijk}),\text{s.t.}:\\
&D\leqslant\sum_{k=1:2}y_{ijk}^2,\\
&-y_{ijk}-\eta_{ijk}\leqslant x_{ik}-x_{jk}\leqslant 1-y_{ijk},\\
&-1+y_{ijk}\leqslant x_{ik}-x_{jk}\leqslant y_{ijk}+\eta_{ijk},\\
&z_{ijk}+\eta_{ijk}\leqslant y_{ijk}\leqslant -z_{ijk}, \qquad\qquad \text{(FTPP)}\\
&z_{ijk}\leqslant x_{ik}-x_{jk}\leqslant z_{ijk}+2\zeta_{ijk},\\
&-z_{ijk}-2\Big(1-\zeta_{ijk}\Big)\leqslant x_{ik}-x_{jk}\leqslant -z_{ijk},\\
&0\leqslant x_{ik}\leqslant 1, y_{ijk}\in\mathbb{R}, z_{ijk}\in\mathbb{R}, \eta_{ijk}\in\{0,1\}, \zeta_{ijk}\in\{0,1\}.
\end{aligned}$$

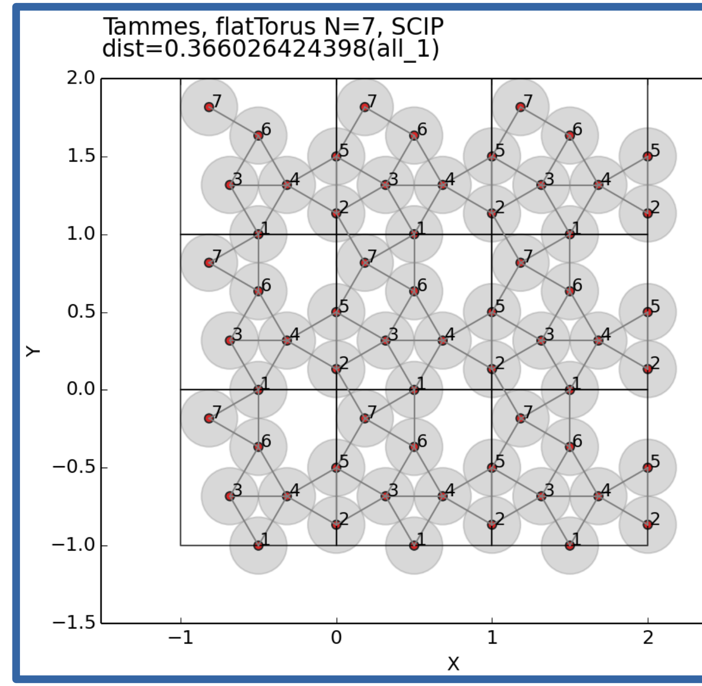
FTPP N=7, three solutions, SCIP only

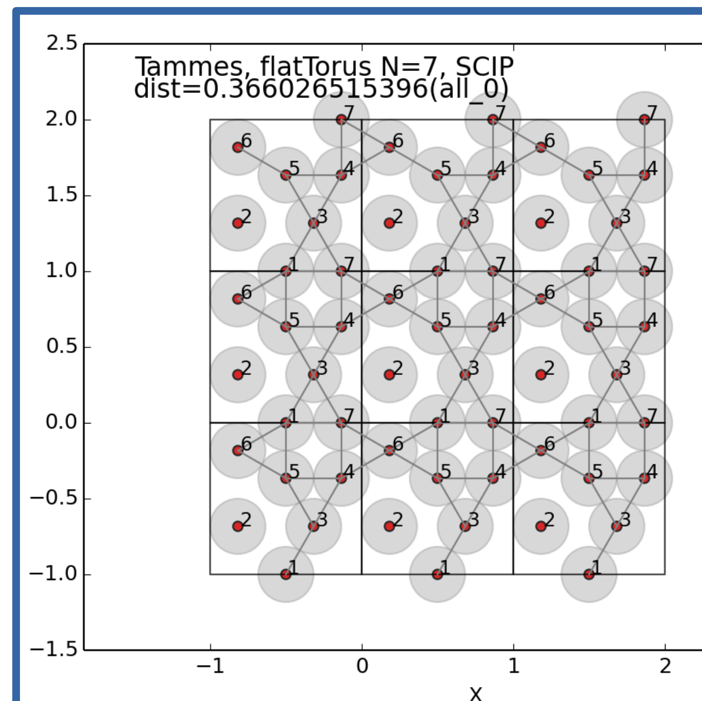
The case is **N=7, 2550 sec (43 min) by standalone SCIP**, gives three different (up to isometric transformation) solutions

0.366026...

$$d^{opt} = \frac{1}{1+\sqrt{3}}$$



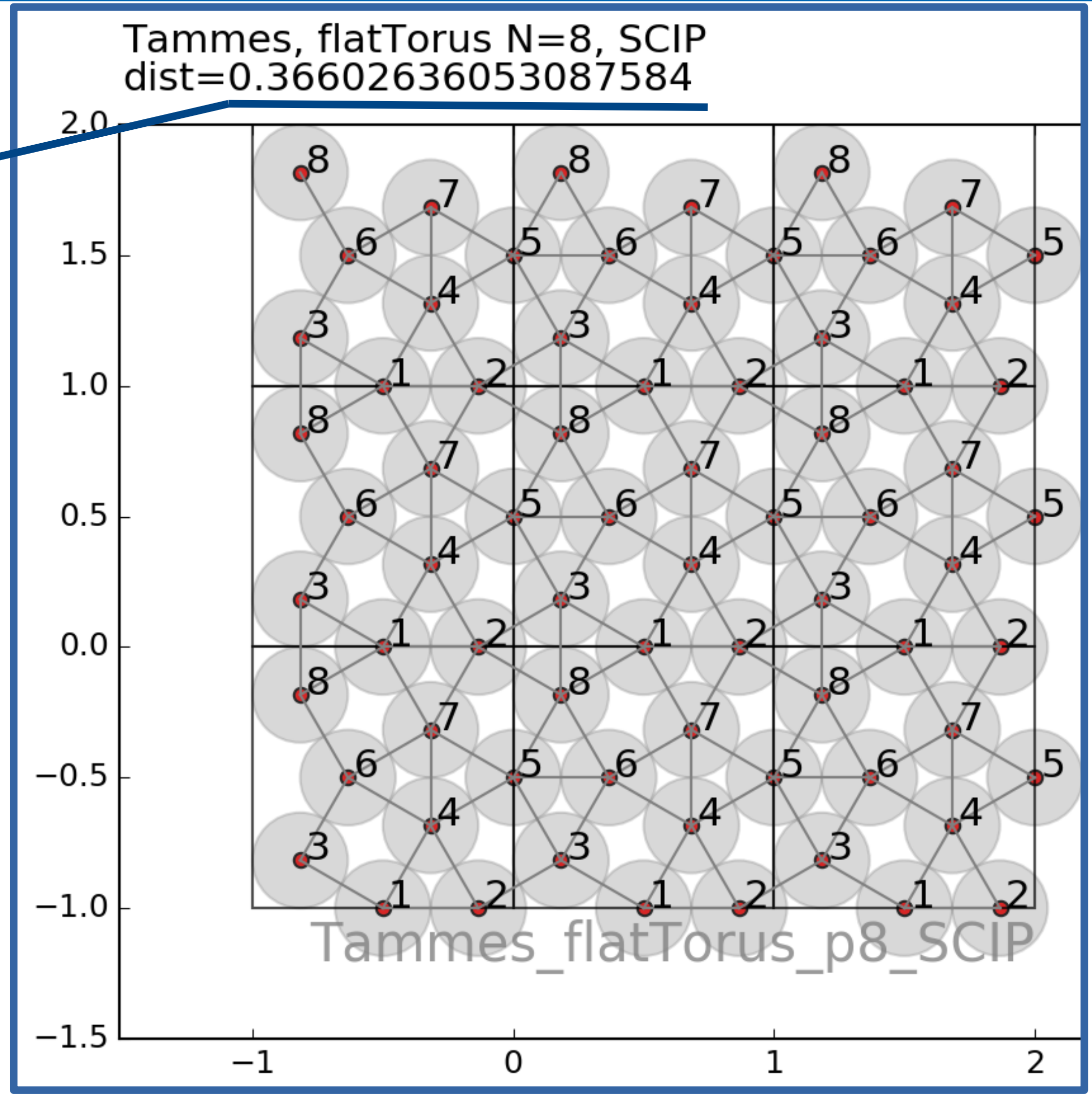




FTPP, N=8, SCIP vs ParaSCIP NRC "Kurchatov institute"

0.366026...

$$d^{opt} = \frac{1}{1+\sqrt{3}}$$



SCIP, 1CPU, 1 solver
780 min
454 min, additional constraint

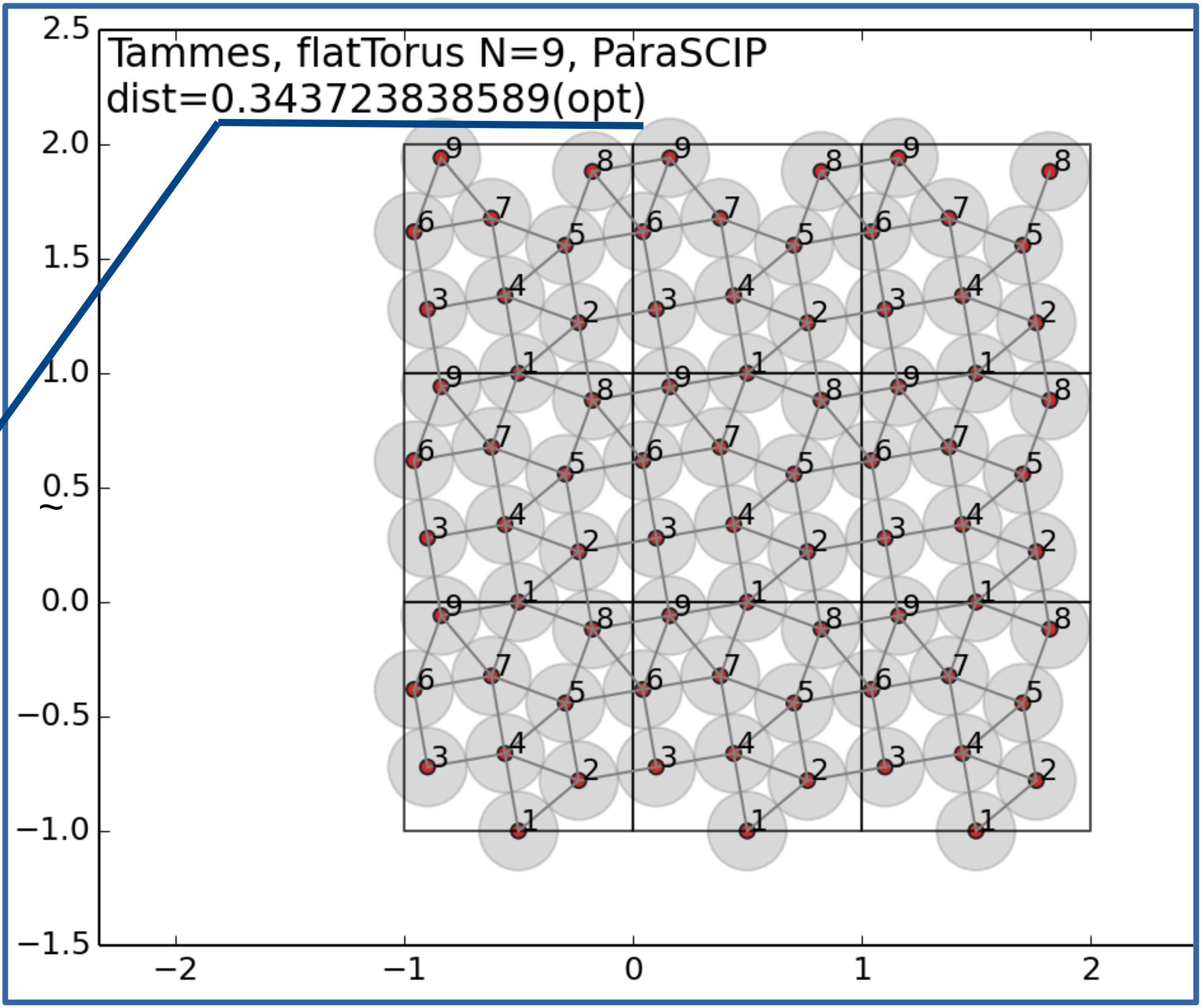
ParaSCIP, 8 CPUs, 7 solvers
126 min
Efficiency (CPU): 780/126/8 = 0.77
Efficiency (solvers): 780/126/7 = 0.88

FTPP N=9, ParaSCIP, NRC "Kurchatov institute"

ParaSCIP,
128 CPUs, 127 solvers
956 min ~ 16 hours

16*128 ~ 2000 CPU*h

0.3437238385...



$$d^{conj} = \frac{1}{\sqrt{5+2\sqrt{3}}}$$

Numerical confirmation of conjecture from the article (as analytic formula): Oleg Musin, Anton Nikitenko. Optimal packings of congruent circles on a square flat torus. *Discrete & Computational Geometry*, 55(1):1–20, 2016.

Conclusion

DDBNB, https://github.com/distcomp/ddbnb

- **Proposed approach, though very simple, may be useful if domain decomposition has been done properly**

!! Way of decomposition remains an open issue !!

Only informal, inexact reasoning yet.

- **DDBNB Everest application became rather mature and provides use of domain decomposition and/or concurrent mode of BnB parallelization in heterogeneous computing environment**

- **Main drawback of DDBNB is absence of dynamic workload balance between available resources**

Future plans

- To integrate ParaSCIP in DDBNB to unite different clusters in solving one problem.
- To try Flat Torus Packing problem with circles of different diameters.
- To try our approach for molecular clusters conformation problems minimizing, i.e. Lennard-Jones potential (may be reduced for problem with polynimials)

And we are open for collaboration, http://distcomp.ru

Thank you for your attention.

Questions?

ParaSCIP, Flat Torus Packing N=7, 9

Our success story with ParaSCIP:

1. Running on HPC4/HPC5, NRC "Kurchatov Institute", ~22 000 cores, T-Platforms (458 in World Top500, 4 in Russia Top50)

KIAE has CentOS 6 with GCC 4.4 and doesn’t support C++11 extensions required by SCIP.
Workaround: take another host with a similar CentOS version and devtoolset-7 (GCC 7.3); build solvers; copy them to the cluster.
It might give not optimal code due to difference in CPU architecture (unknown to compiler!)

2. Computer aided confirmation of one open problem in Combinatorial Geometry: Packing Flat Torus with N=9 congruent circles.