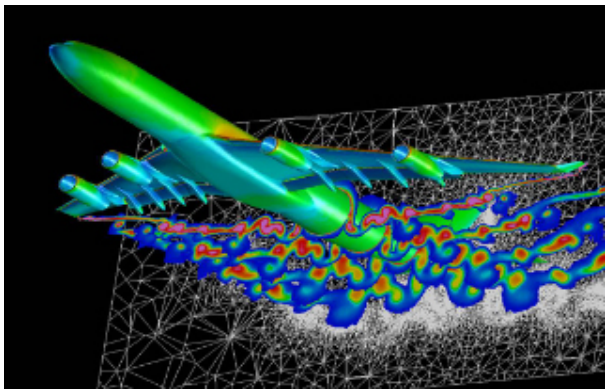
Topology-aware resource management for HPC applications

VILLIERMET Adèle
JEANNOT Emmanuel
MERCIER Guillaume
TADaaM team
Inria Bordeaux Sud-Ouest

July 4, 2016

INTRODUCTION: Context

- More computation resources needed
- Parallel architectures
- More and more complex



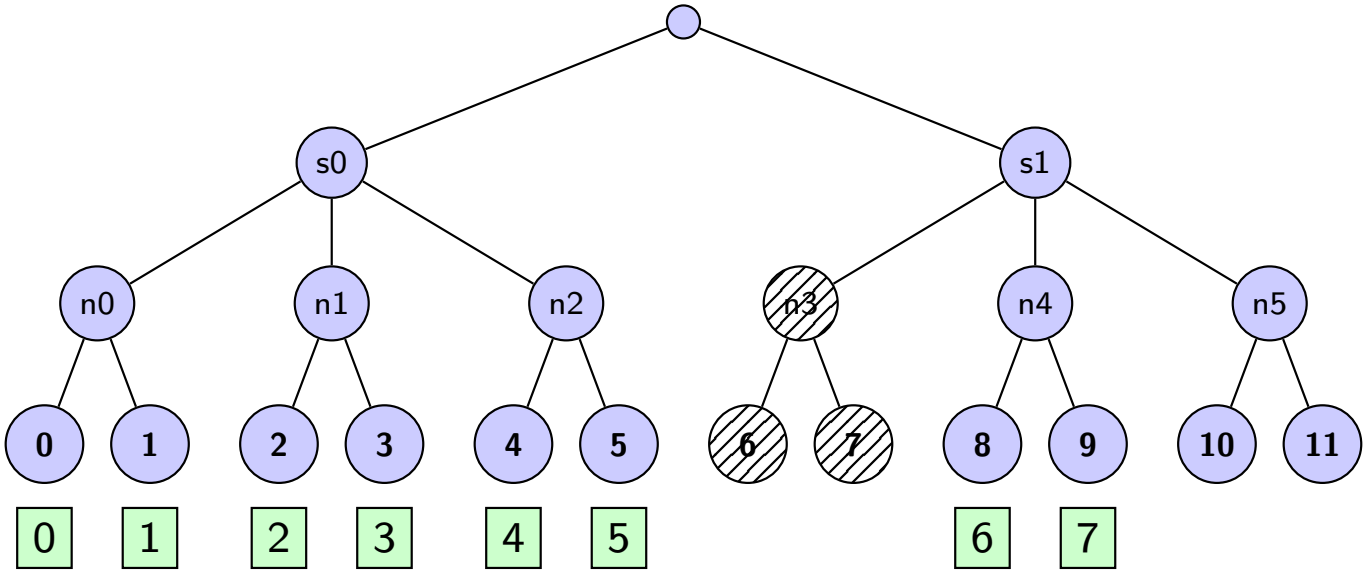
INTRODUCTION: 2 Problematics

- Managing and sharing resources:
  RJMS or Batch Scheduler (SLURM, OAR, PBS, etc.)
  - provided informations (allocations size)
  - partial informations (allocations length)
  - unknown informations (futur jobs)

- Locality and process placement
  - collected informations (network, architecture)
  - applications characteristic (compute bound? memory bound?)

Contents

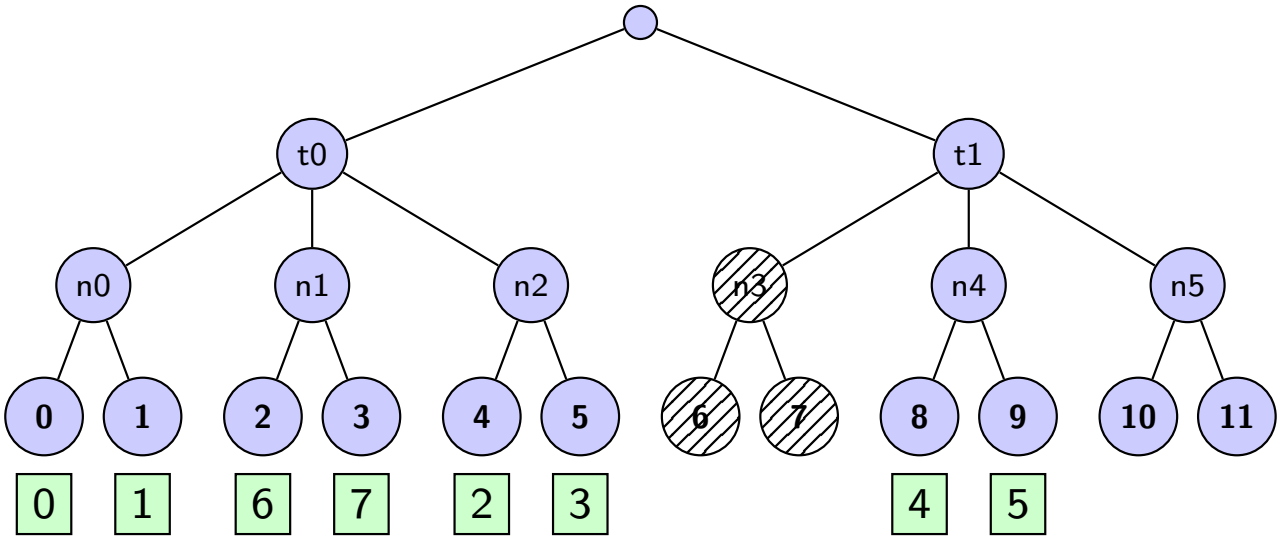
SLURM: Simple Linux Utility for Resource Management



TreeMatch: process placement

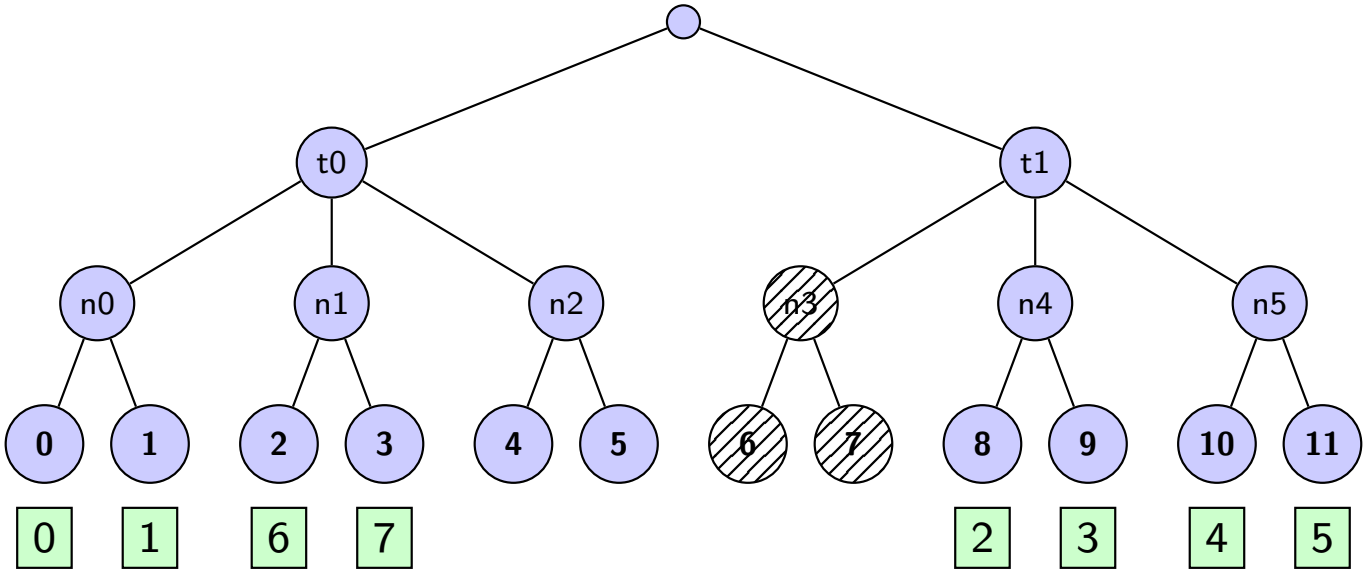
Processus	0-1	2-3	4-5	6-7
0-1	0	20	0	2000
2-3	20	0	1000	0
4-5	0	1000	0	10
6-7	2000	0	10	0

$$\text{Hop-Byte}(\sigma) = \sum_{1 \leq i < j \leq n} \omega(i,j) \times d(\sigma(i), \sigma(j))$$



Contribution: TreeMatch integration within SLURM

Processus	0-1	2-3	4-5	6-7
0-1	0	20	0	2000
2-3	20	0	1000	0
4-5	0	1000	0	10
6-7	2000	0	10	0



Contribution: TreeMatch integration within SLURM

Information provided

- Communication matrix:
  srun -m TREEMATCH=/communication/matrix/path
- Cluster topology:
  global topology + constraints due to other jobs

Alternative method with subtree
→ reduce TreeMatch overhead

Experimental Validation

- Edel cluster from the Grid'5000 Grenoble site
- 72 nodes with 2 Intel Xeon E5520 CPUs (2.27 GHz, 4 cores/cpu) and 24GB of memory
- Based on a Curie (TGCC cluster, 80640 cores) workload trace
- Communication matrices randomly generated

Experimental Validation

Need to modify the jobs run times dynamically according to their allocation

- compute 2 allocations: SLURM and TreeMatch
- R: ratio between their hop-byte
- α: ratio of communication
- $T = T_{calc} + T_{comm}$
  $T_{comm} = \alpha T$

$$\begin{aligned} T' &= T_{calc} + RT_{comm} \\ &= R\alpha T + (1-\alpha)T \\ &= (1 + R\alpha - \alpha)T \end{aligned}$$

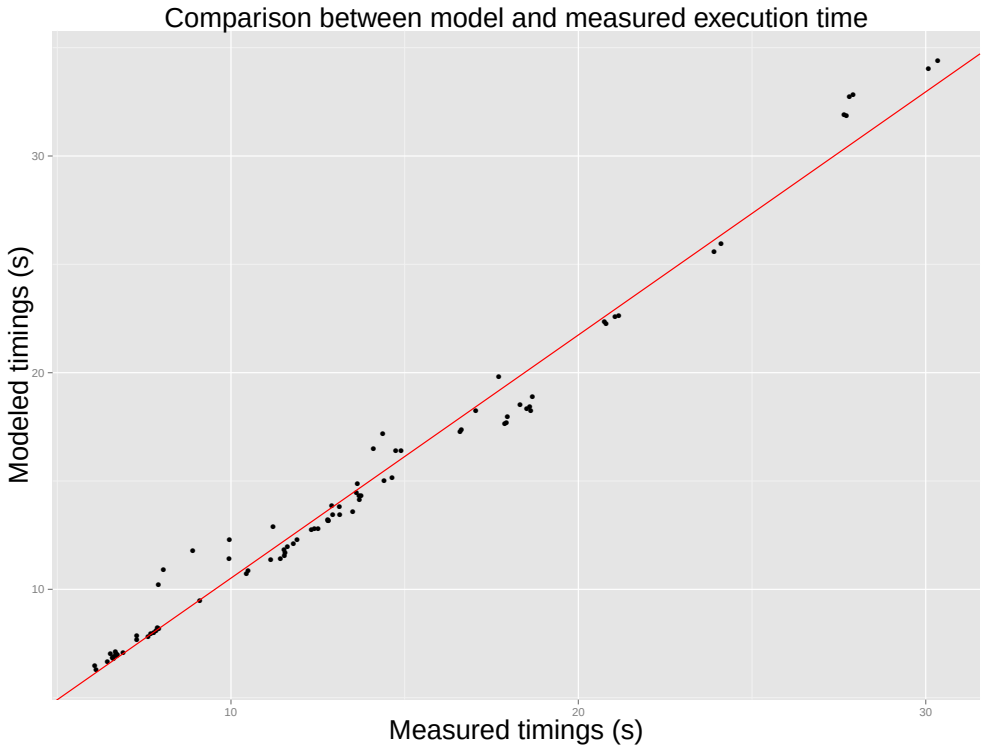


Figure : Comparison for the minighost application with a communication ratio between 5% and 45%

Experimental Validation: emulation

Com	SLURM	TM-A	TM-Isub	TM-I
50%	8318	6407	6073	6077
33%	8316	7502	6821	6887

(a) Makespan

Com	SLURM	TM-A	TM-Isub	TM-I
50%	33%	42%	44%	44%
33%	33%	36%	40%	39%

(b) Utilization

Figure : Workload Metrics for the different strategies and different amount of communication ratio

Experimental Validation: emulation

33% of communication

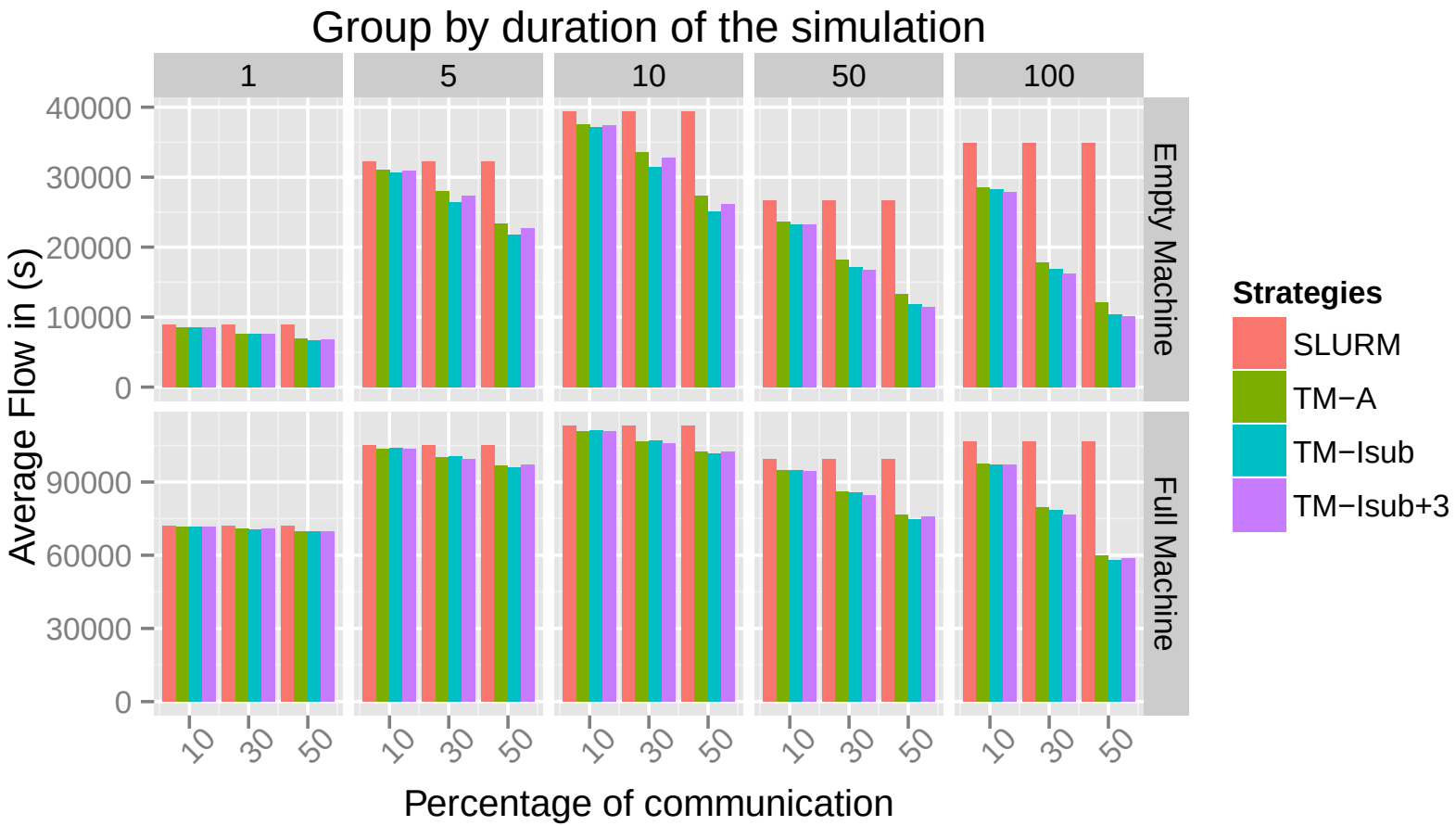
SLURM	−205.85 s / 0.91	−183.35 s / 1.06	−167.65 s / 1.31
[−253 s, −20 s]	TM−Isub	22.50 s / 1.16	38.20 s / 1.44
[−213 s, −6 s]	[5 s, 18 s]	TM−I	15.70 s / 1.24
[−185 s, −4 s]	[14 s, 20 s]	[7 s, 13 s]	TM−A

50% of communication

SLURM	−322.23 s / 0.79	−312.03 s / 0.94	−274.40 s / 1.16
[−396 s, −27 s]	TM−Isub	10.20 s / 1.19	47.83 s / 1.47
[−306 s, −13 s]	[4 s, 14 s]	TM−I	37.63 s / 1.24
[−307 s, −11 s]	[12 s, 23 s]	[3 s, 11 s]	TM−A

Figure : Statistical comparison of selection methods: flowtime

Experimental Validation: simulation



Conclusion

- New allocation policy
- Better results for global and user metrics

Future works:

- Include within official future SLURM releases
- Fragmentation
- Communication pattern accuracy

Thanks

INRIA-Bordeaux
TADaaM team