My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

My historical perspective

- I made it to the 8 CCGSC workshops!
- I talked about a nice little scheduling problem in 1992
- I talked about a nice little scheduling problem in 1994
- I talked about a nice little scheduling problem in 1996
- I talked about a nice little scheduling problem in 1998
- I talked about a nice little scheduling problem in 2000
- I talked about a nice little scheduling problem in 2002
- I talked about a nice little scheduling problem in 2004
- I wondered what I should do this year?
- Maybe I should find a nice little scheduling problem! ☺

Did anything change in the meantime?

- Parallel algorithm design and scheduling were already difficult tasks with homogeneous machines
- **On heterogeneous platforms, it gets worse**
- Patrick Geoffray went from kindergarten to Myricom but he's still a kid! ☺
- He says that only embarrassingly parallel applications can be deployed on the grid
- Clearly, he is over optimistic! ☹

Did anything change in the meantime?

- Parallel algorithm design and scheduling were already difficult tasks with homogeneous machines
- **On heterogeneous platforms, it gets worse**
- Patrick Geoffray went from kindergarten to Myricom but he's still a kid! ☺
- He says that only embarrassingly parallel applications can be deployed on the grid
- Clearly, he is over optimistic! ☹

Did anything change in the meantime?

- Parallel algorithm design and scheduling were already difficult tasks with homogeneous machines
- **On heterogeneous platforms, it gets worse**
- Patrick Geoffray went from kindergarten to Myricom but he's still a kid! ☺
- He says that only embarrassingly parallel applications can be deployed on the grid
- Clearly, he is over optimistic! ☹

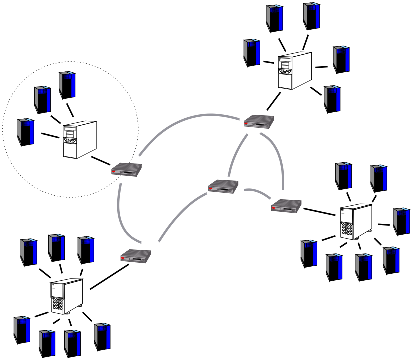
Did anything change in the meantime?

- Parallel algorithm design and scheduling were already difficult tasks with homogeneous machines
- **On heterogeneous platforms, it gets worse**
- Patrick Geoffray went from kindergarten to Myricom but he's still a kid! ☺
- He says that only embarrassingly parallel applications can be deployed on the grid
- Clearly, he is over optimistic! ☹

Did anything change in the meantime?

- Parallel algorithm design and scheduling were already difficult tasks with homogeneous machines
- **On heterogeneous platforms, it gets worse**
- Patrick Geoffray went from kindergarten to Myricom but he's still a kid! ☺
- He says that only embarrassingly parallel applications can be deployed on the grid
- Clearly, he is over optimistic! ☹

A nice little embarrassingly parallel application



- One (divisible load) application running on each cluster
  ⇒ Which fraction of the job to delegate to other clusters?
- Different communication-to-computation ratios
  ⇒ How to ensure fair scheduling and good resource utilization?

The great talk you've been expecting

Revisiting matrix product on heterogeneous platforms

Jack Dongarra, Zhiao Shi, UT Knwoville

Jean-François Pineau, Yves Robert, Frédéric Vivien, ENS Lyon

The great talk you've been expecting

Revisiting matrix product on heterogeneous platforms

Eh wait!

Experiments are not ready?!

Jean-François Pineau, Yves Robert, Frédéric Vivien, ENS Lyon

Scheduling and Data Redistribution Strategies on Star Platforms

Loris Marchal, Veronika Rehn,
Yves Robert and Frédéric Vivien

GRAAL team, LIP
École Normale Supérieure de Lyon

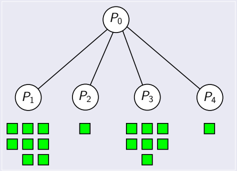
September 2006

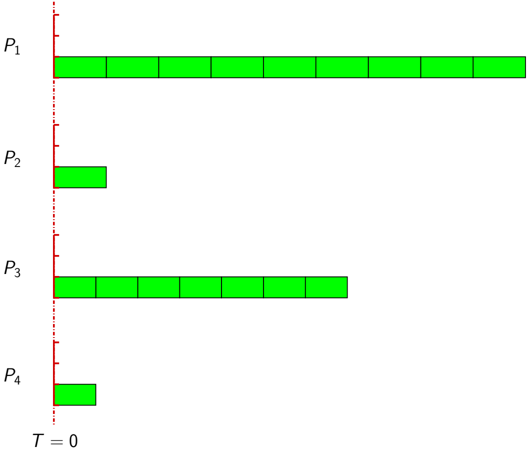
Outline

Outline

Example

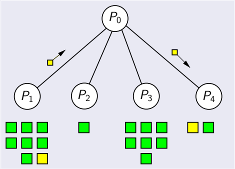
Architecture

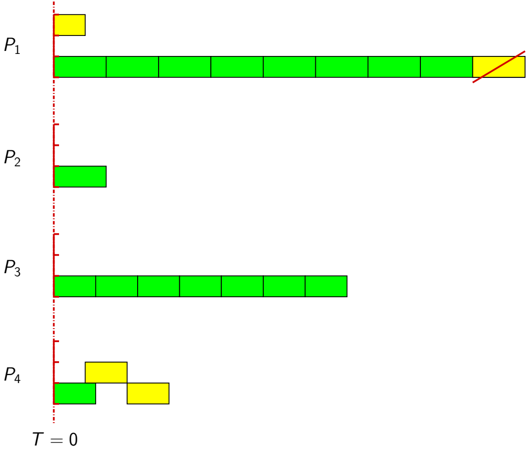




Example

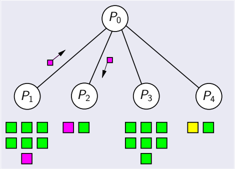
Architecture

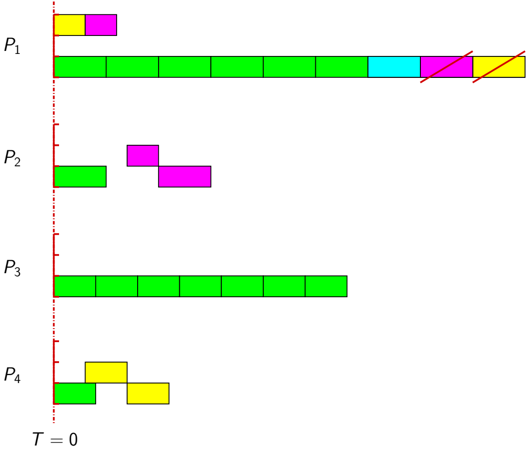




Example

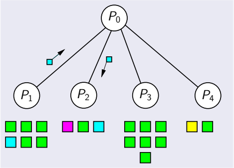
Architecture

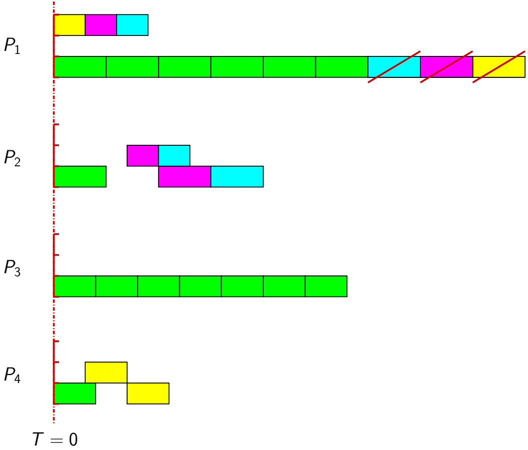




Example

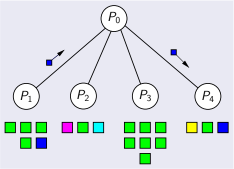
Architecture

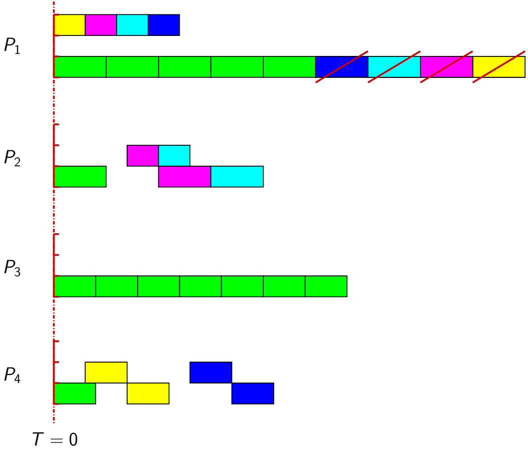




Example

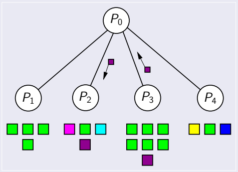
Architecture

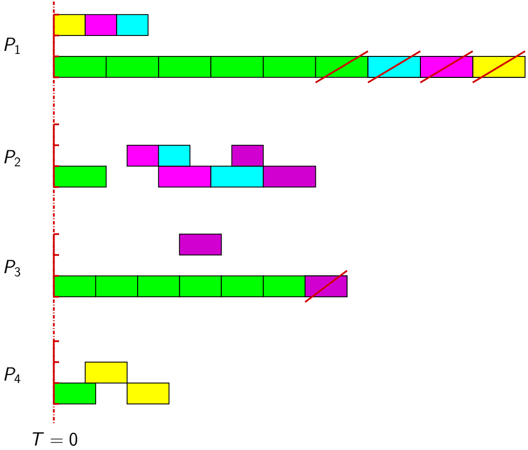




Example

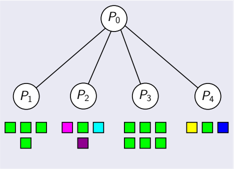
Architecture

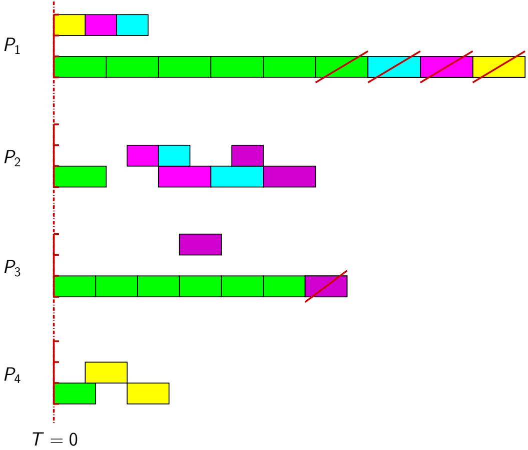




Example

Architecture





Framework

- Master-slave platforms
- New: Distributed loads

Problem

Redistribution of data

Goal: Minimize overall processing time

Data models

- Independent tasks
- Divisible loads

Framework

- Master-slave platforms
- New: Distributed loads

Problem

Redistribution of data

Goal: Minimize overall processing time

Data models

- Independent tasks
- Divisible loads

Framework

- Master-slave platforms
- New: Distributed loads

Problem

Redistribution of data

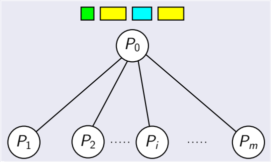
Goal: Minimize overall processing time

Data models

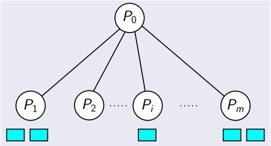
- Independent tasks
- Divisible loads

Related Work

Known results



Our approach



Independent tasks

- Application: BOINC (e.g. Einstein@home)
- NP-completeness for different task sizes

Divisible load theory

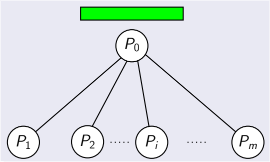
- Perfect parallel jobs
- Optimal algorithms for video processing (Altilar, Paker)

Redistribution algorithms

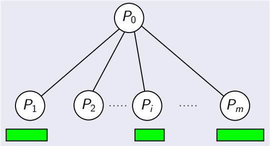
- NP-completeness (Kremer)
- Optimality for particular cases: homogeneous ring topologies

Related Work

Known results



Our approach



Independent tasks

- Application: BOINC (e.g. Einstein@home)
- NP-completeness for different task sizes

Divisible load theory

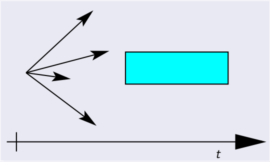
- Perfect parallel jobs
- Optimal algorithms for video processing (Altilar, Paker)

Redistribution algorithms

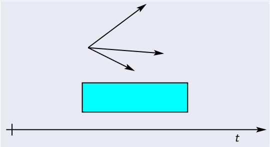
- NP-completeness (Kremer)
- Optimality for particular cases: homogeneous ring topologies

Related Work

Known results



Our approach



Independent tasks

- Application: BOINC (e.g. Einstein@home)
- NP-completeness for different task sizes

Divisible load theory

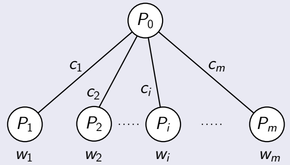
- Perfect parallel jobs
- Optimal algorithms for video processing (Altilar, Paker)

Redistribution algorithms

- NP-completeness (Kremer)
- Optimality for particular cases: homogeneous ring topologies

Model

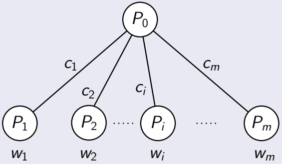
Star network



- Star network $S = P_0, P_1, \ldots, P_m$
- Communication cost c_i
- Computing power w_i
- Initial data L_i
- Independent and identical tasks
- Linear cost model
- Bidirectional one-port model
- Objective function:
  Minimize makespan

Model

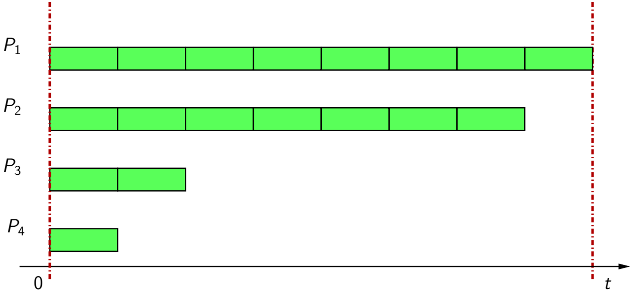
Star network



- Star network $S = P_0, P_1, \ldots, P_m$
- Communication cost c_i
- Computing power w_i
- Initial data L_i
- Independent and identical tasks
- Linear cost model
- Bidirectional one-port model
- Objective function:
  Minimize makespan

Best-Balance Algorithm - BBA

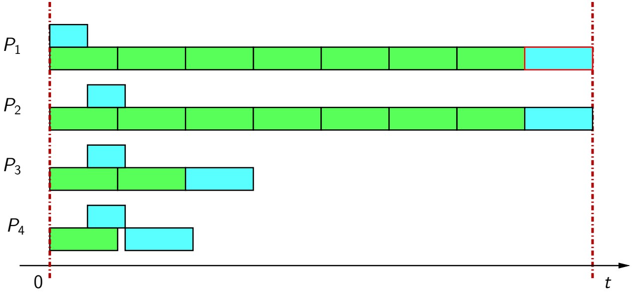
- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

Best-Balance Algorithm - BBA

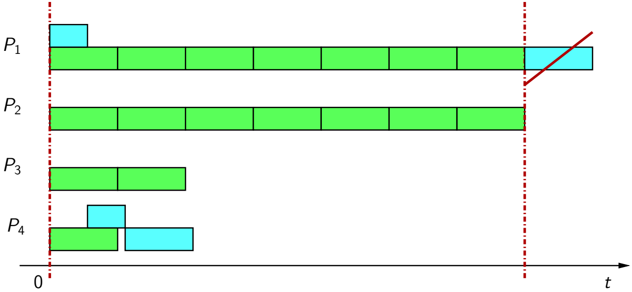
- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

Best-Balance Algorithm - BBA

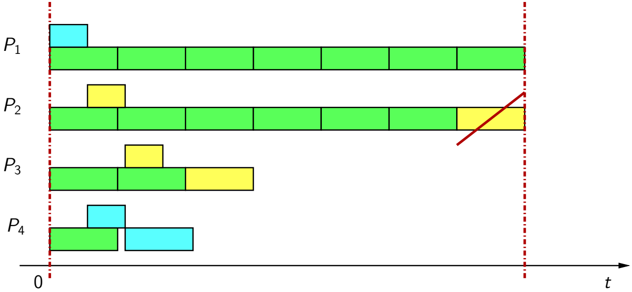
- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

Best-Balance Algorithm - BBA

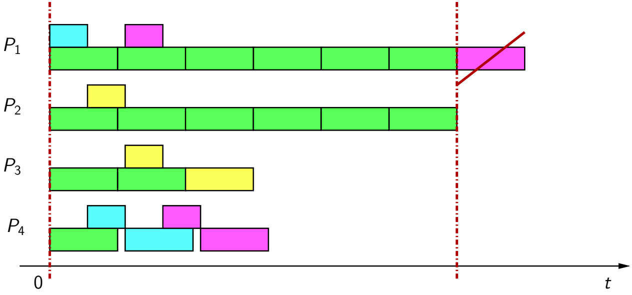
- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

Best-Balance Algorithm - BBA

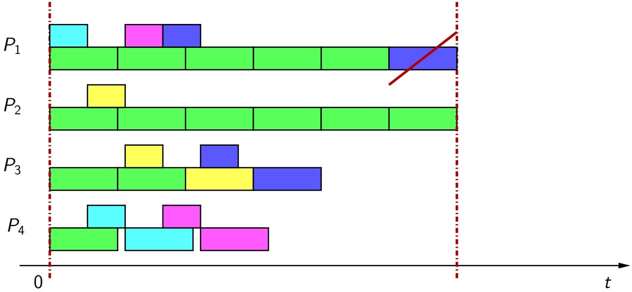
- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

Best-Balance Algorithm - BBA

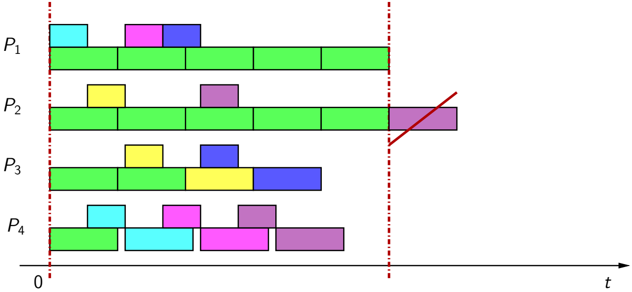
- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

Best-Balance Algorithm - BBA

- Homogeneous communication links
- Homogeneous workers



Principle: Local optimization of current makespan

BBA- Optimality

Theorem

BEST-BALANCE ALGORITHM *calculates an optimal schedule S on a fully homogeneous star network.*

Moore-Based Binary-Search Algorithm - MBBSA

- Homogeneous communication links
- Heterogeneous workers
- Makespan M

Principle:

- Moore's algorithm
- Schedule within M
- Binary search

Moore's algorithm

```
Order the jobs by non-decreasing deadlines: d_1 ≤ d_2 ≤ ··· ≤ d_d
σ ← ∅; t ← 0
for i := 1 to n do
    σ ← σ ∪ {i}
    t ← t + w_i
    if t > d_i then
        Find job j in σ with largest w_j value
        σ ← σ\{j}
        t ← t − w_j
    end if
end for
```

Moore-Based Binary-Search Algorithm - MBBSA

- Homogeneous communication links
- Heterogeneous workers
- Makespan M

Principle:

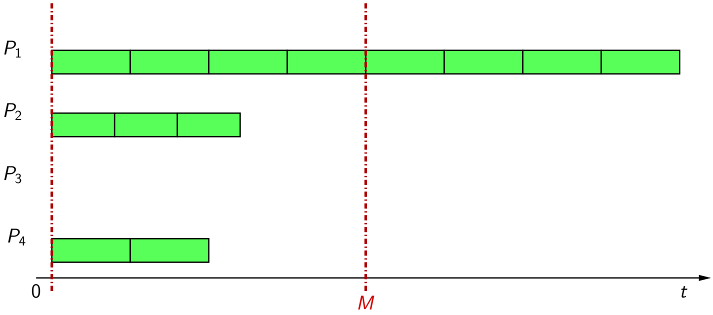
- Moore's algorithm
- Schedule within M
- Binary search

Moore's algorithm

```
Order the jobs by non-decreasing deadlines: d_1 ≤ d_2 ≤ ··· ≤ d_d
σ ← ∅; t ← 0
for i := 1 to n do
    σ ← σ ∪ {i}
    t ← t + w_i
    if t > d_i then
        Find job j in σ with largest w_j value
        σ ← σ\{j}
        t ← t − w_j
    end if
end for
```

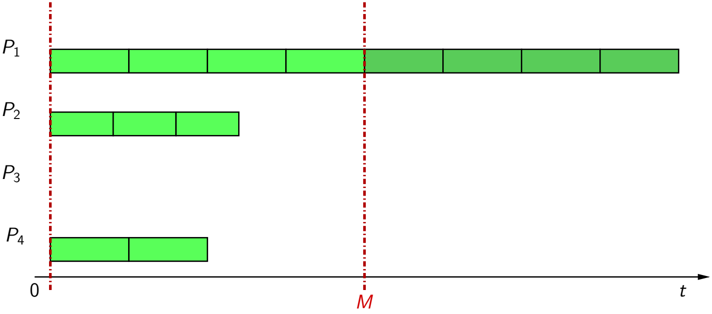
MBBSA- Phase 1 + 2

Determination of senders and receivers



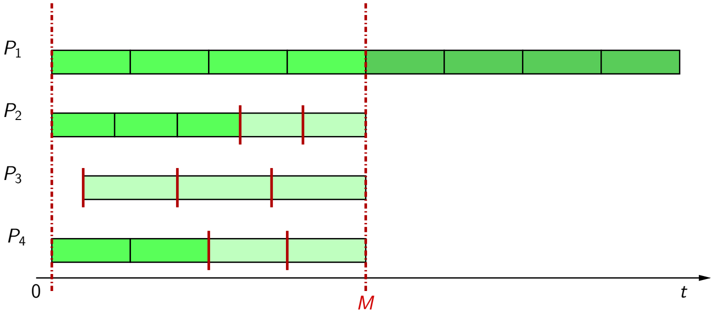
MBBSA- Phase 1 + 2

Determination of senders and receivers



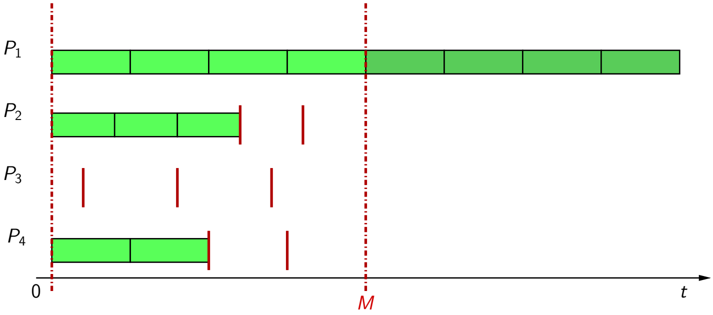
MBBSA- Phase 3

Computation of deadlines



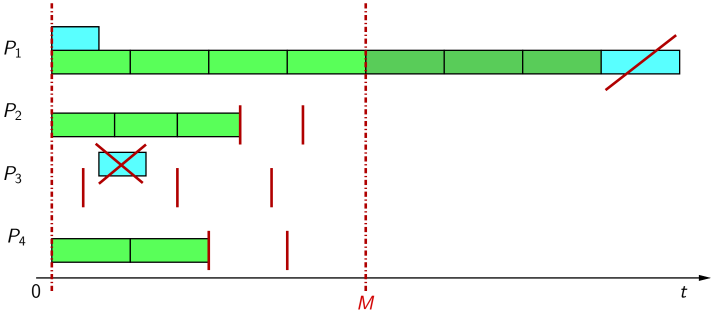
MBBSA- Phase 4

Scheduling step



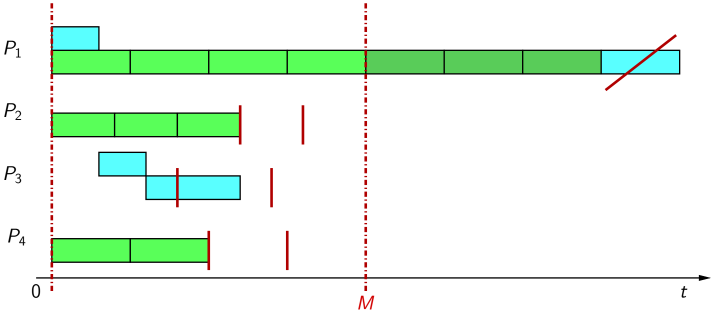
MBBSA- Phase 4

Scheduling step



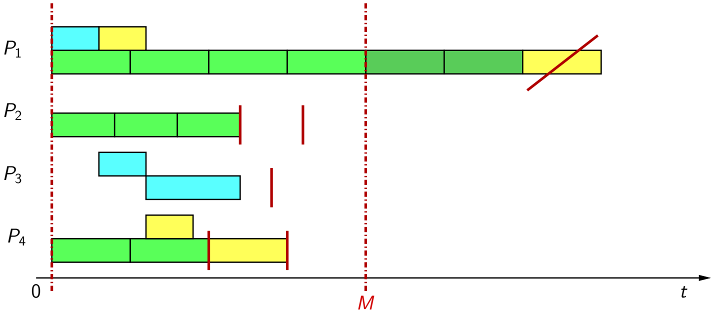
MBBSA- Phase 4

Scheduling step



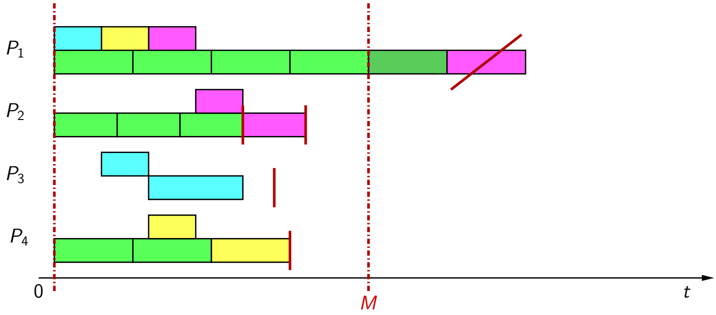
MBBSA- Phase 4

Scheduling step



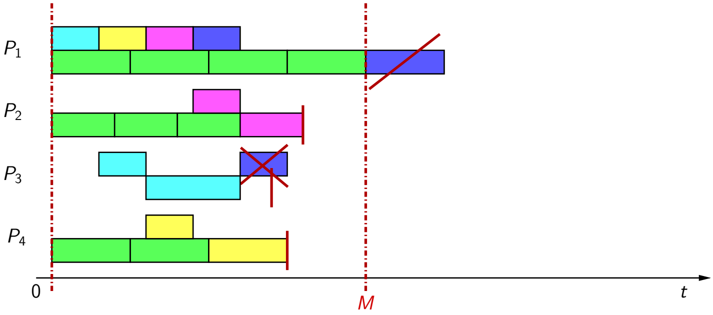
MBBSA- Phase 4

Scheduling step



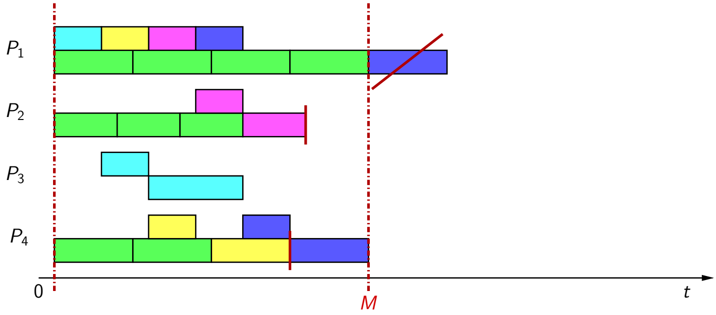
MBBSA- Phase 4

Scheduling step



MBBSA- Phase 4

Scheduling step



MBBSA- Optimality

Theorem

(i) *MBBSA succeeds to build a schedule σ for a given makespan M, if and only if there exists one.*

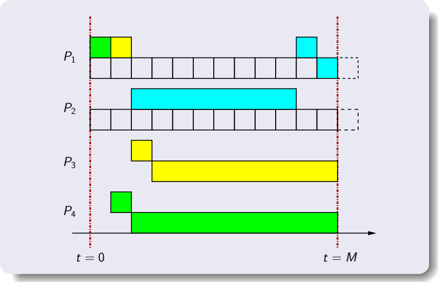
(ii) *Binary search algorithm returns in polynomial time an optimal schedule σ for bus platforms (homogeneous communication links and heterogeneous workers).*

Dealing with fully heterogeneous platforms

Difficulty: Who is sender, who is receiver?

M = 12

Worker	c	w	load
P_1	1	1	13
P_2	8	1	13
P_3	1	9	0
P_4	1	10	0



NP-completeness

Scheduling Problem for Master-Slave Tasks on a Star of Heterogeneous Processors

Definition (SPMSTSHP)

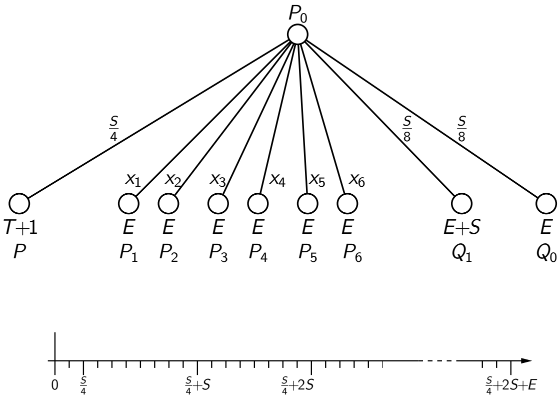
Let N be a star-network. Let T be a deadline.
"Is it possible to redistribute tasks and process them in time T?".

Theorem

NP-complete in the strong sense.

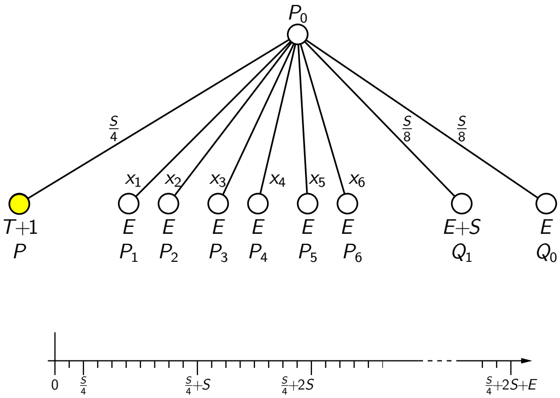
Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



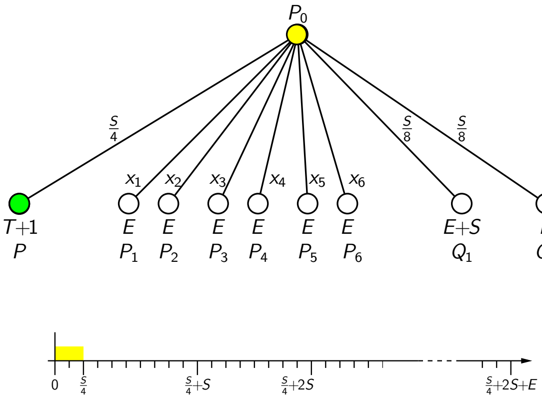
Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



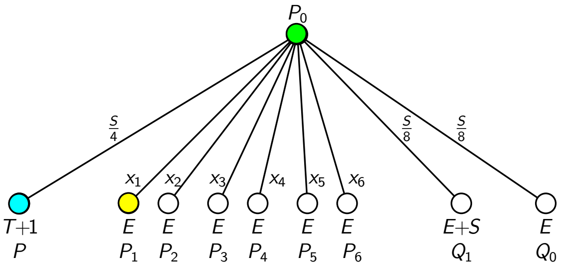
Proof: Reduction to 3-partition

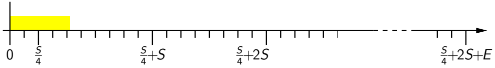
Proof: Reduction to 3-partition problem



Proof: Reduction to 3-partition

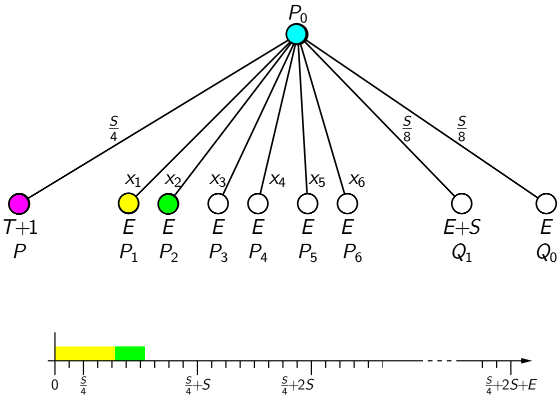
Proof: Reduction to 3-partition problem





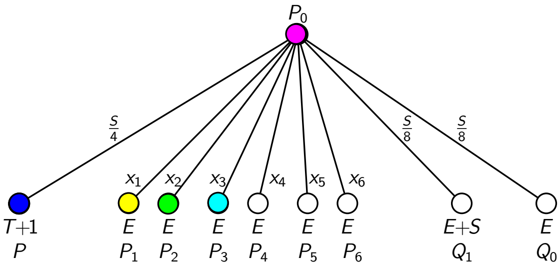
Proof: Reduction to 3-partition

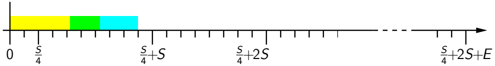
Proof: Reduction to 3-partition problem



Proof: Reduction to 3-partition

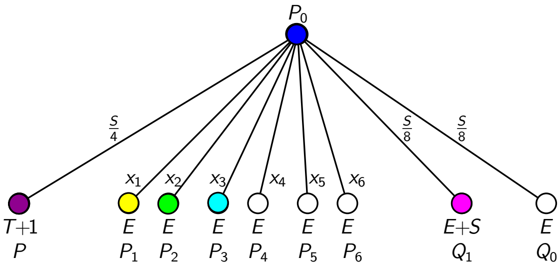
Proof: Reduction to 3-partition problem

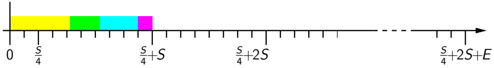




Proof: Reduction to 3-partition

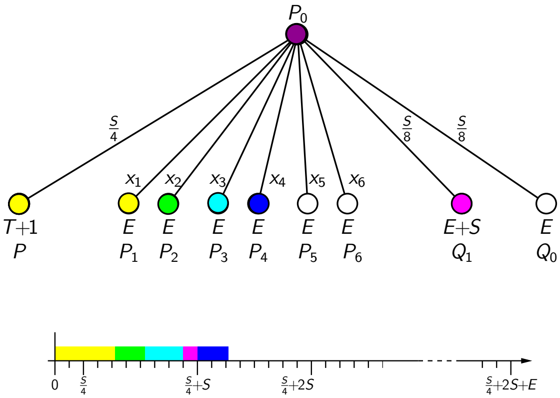
Proof: Reduction to 3-partition problem





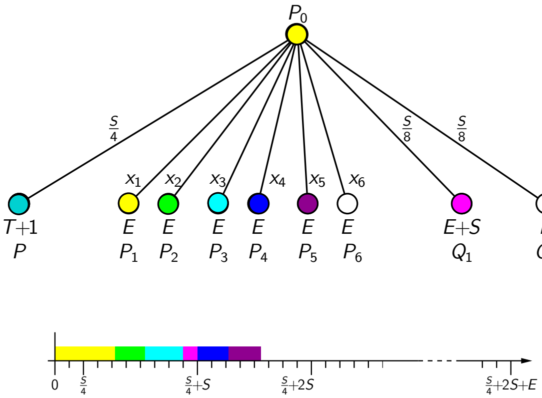
Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



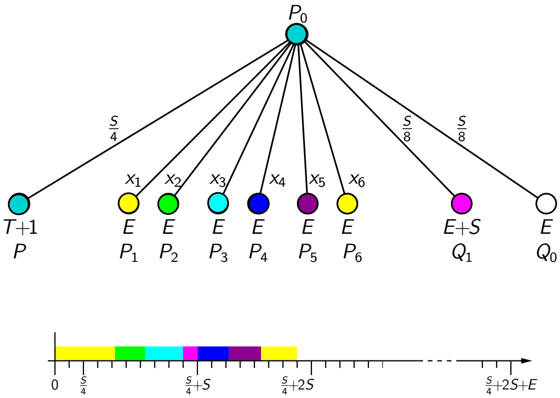
Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



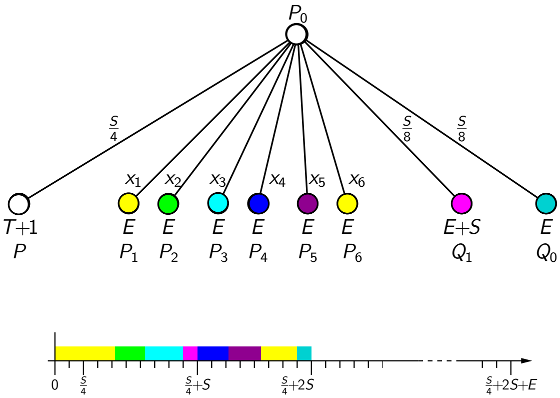
Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



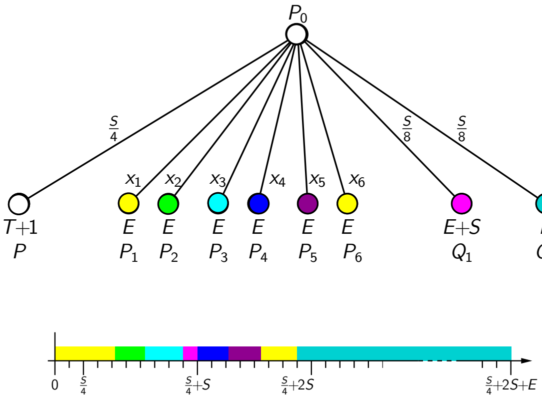
Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



Proof: Reduction to 3-partition

Proof: Reduction to 3-partition problem



Impact of Heterogeneity

Platform type: Comm.	Platform type: Comp.	Difficulty
Hom.	Hom.	simple greedy algorithm
Hom.	Het.	complicated algorithm
Het.	Hom.	?
Het.	Het.	NP-strong

Heuristics

- BBA
- MBBSA
- R-BSA: Reversed-Binary Search Algorithm
  Combination of greedy algorithm and binary search

Impact of Heterogeneity

Platform type		
Comm.	Comp.	Difficulty
Hom.	Hom.	simple greedy algorithm
Hom.	Het.	complicated algorithm
Het.	Hom.	?
Het.	Het.	NP-strong

Heuristics

- BBA
- MBBSA
- R-BSA: Reversed-Binary Search Algorithm
  Combination of greedy algorithm and binary search

Impact of Heterogeneity

Platform type		
Comm.	Comp.	Difficulty
Hom.	Hom.	simple greedy algorithm
Hom.	Het.	complicated algorithm
Het.	Hom.	?
Het.	Het.	NP-strong

Heuristics

- BBA
- MBBSA
- R-BSA: Reversed-Binary Search Algorithm
  Combination of greedy algorithm and binary search

Impact of Heterogeneity

Platform type		
Comm.	Comp.	Difficulty
Hom.	Hom.	simple greedy algorithm
Hom.	Het.	complicated algorithm
Het.	Hom.	?
Het.	Het.	NP-strong

Heuristics

- BBA
- MBBSA
- R-BSA: Reversed-Binary Search Algorithm
  Combination of greedy algorithm and binary search

Impact of Heterogeneity

Platform type: Comm.	Platform type: Comp.	Difficulty
Hom.	Hom.	simple greedy algorithm
Hom.	Het.	complicated algorithm
Het.	Hom.	?
Het.	Het.	NP-strong

Heuristics

- BBA
- MBBSA
- R-BSA: Reversed-Binary Search Algorithm
  Combination of greedy algorithm and binary search

Outline

Simulations

SimGrid:

Simulator for distributed applications

- 4 platform types
- 1000 instances
- 10 workers
- Random variables

- c_i: 1..100
- w_i: 1..100
- L_i: 0..50

Simulations

SimGrid:

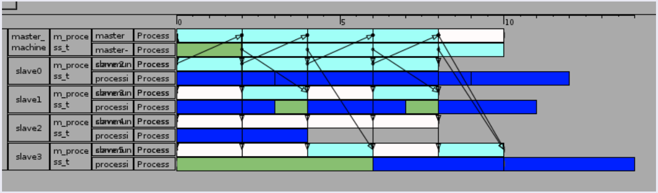
Simulator for distributed applications

- 4 platform types
- 1000 instances
- 10 workers
- Random variables

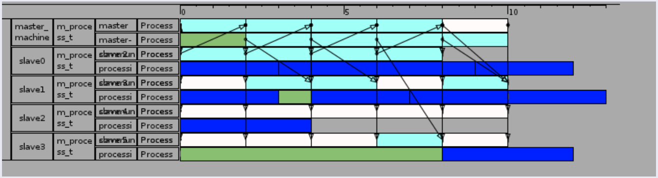
- c_i: 1..100
- w_i: 1..100
- L_i: 0..50

Trace Tests

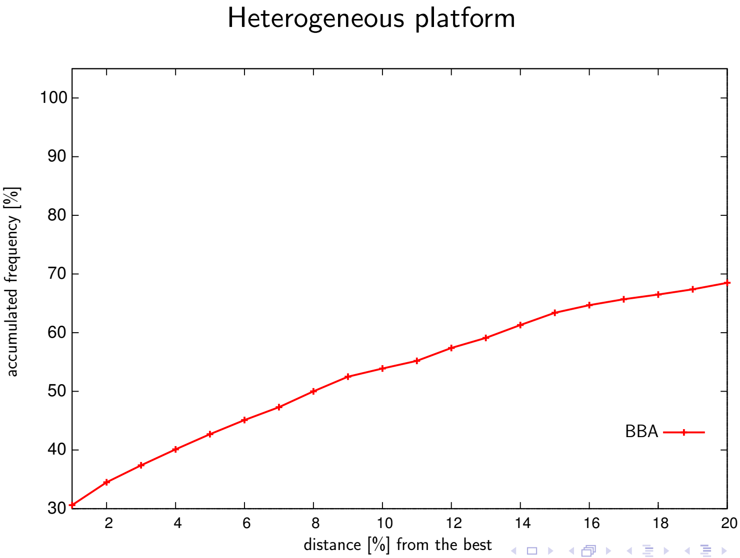
BBA



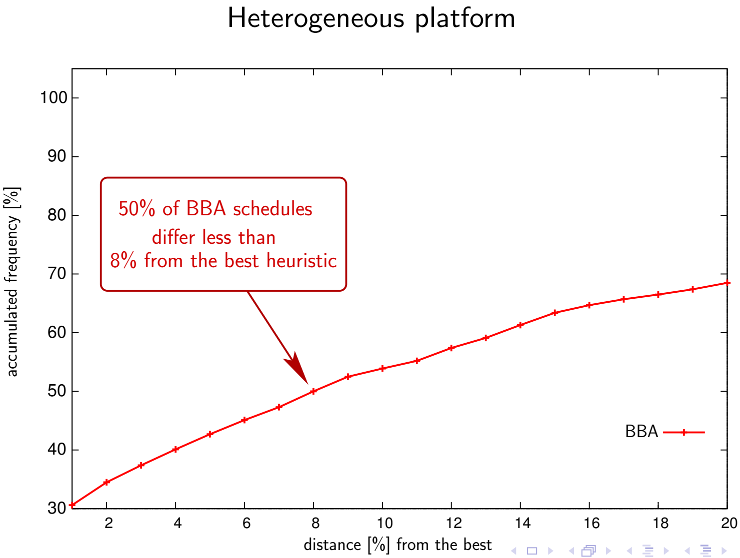
MBBSA



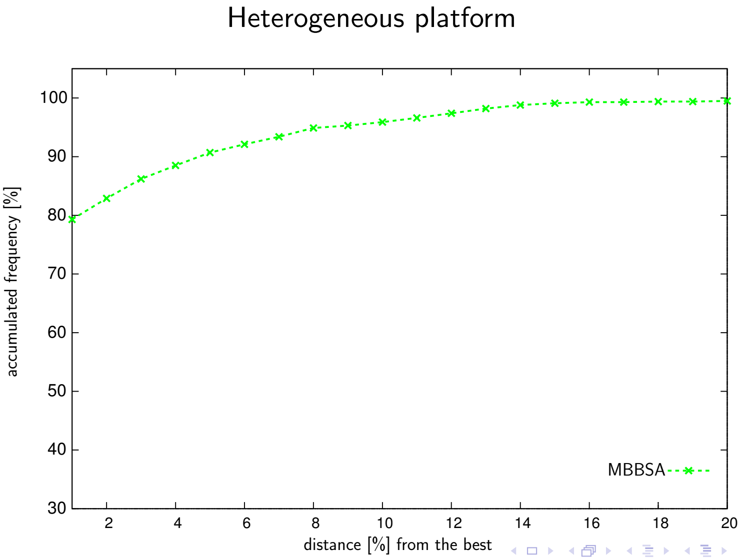
Distance from the Best Heuristic



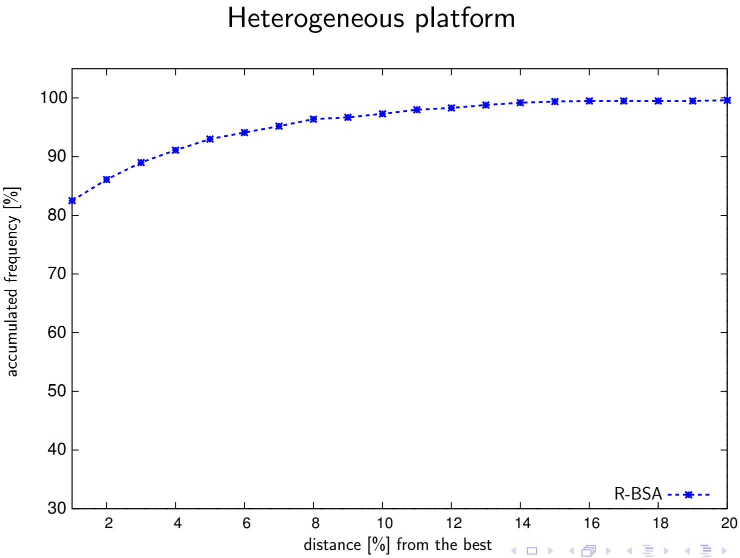
Distance from the Best Heuristic



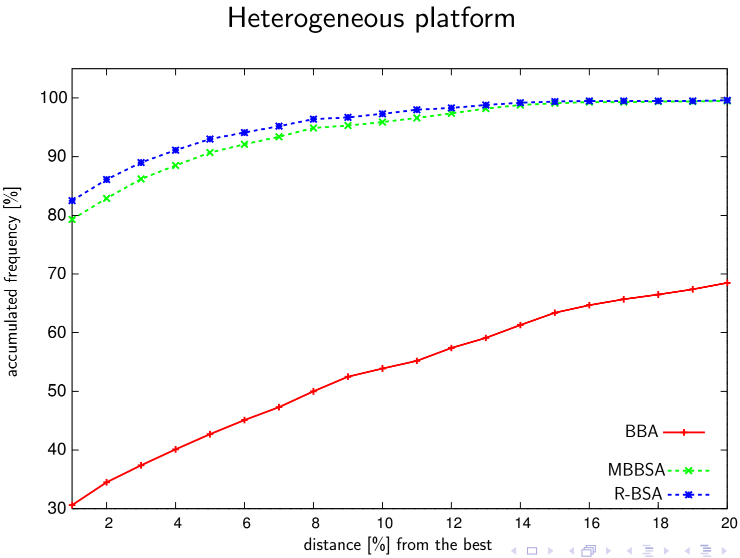
Distance from the Best Heuristic



Distance from the Best Heuristic



Distance from the Best Heuristic



Standard Deviation

Platform type		Standard deviation		
Comm.	Comp.	BBA	MBBSA	R-BSA
Hom	Hom	0	0	0.0107
Hom	Het	0.0006	0	0.0181
Het	Hom	0.4007	0.0208	0.0173
Het	Het	0.3516	0.0327	0.0284

Standard Deviation

Platform type		Standard deviation		
Comm.	Comp.	BBA	MBBSA	R-BSA
Hom	Hom	0	0	0.0107
Hom	Het	0.0006	0	0.0181
Het	Hom	0.4007	0.0208	0.0173
Het	Het	0.3516	0.0327	0.0284

Standard Deviation

Platform type		Standard deviation		
Comm.	Comp.	BBA	MBBSA	R-BSA
Hom	Hom	0	0	0.0107
Hom	Het	0.0006	0	0.0181
Het	Hom	0.4007	0.0208	0.0173
Het	Het	0.3516	0.0327	0.0284

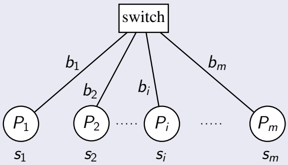
Standard Deviation

Platform type		Standard deviation		
Comm.	Comp.	BBA	MBBSA	R-BSA
Hom	Hom	0	0	0.0107
Hom	Het	0.0006	0	0.0181
Het	Hom	0.4007	0.0208	0.0173
Het	Het	0.3516	0.0327	0.0284

Outline

Framework

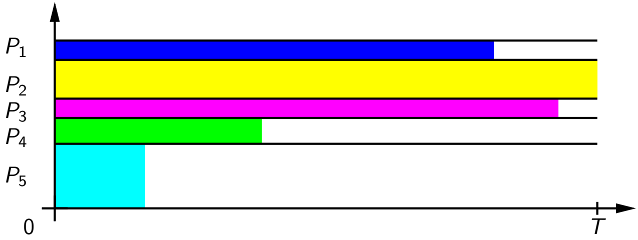
Star network



- Switch as master
- m workers
- Computation speed s_i
- Bandwidth b_i
- Divisible load α_i
- Linear cost model
- Overlapped unbounded switch model

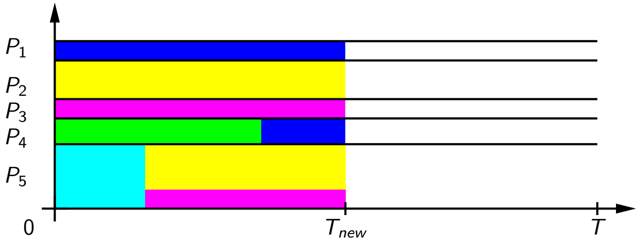
Redistribution Strategy

Goal: Every worker finishes at the same time



Redistribution Strategy

Goal: Every worker finishes at the same time



Solution for Divisible Loads

Imbalance of a worker δ_i

Linear program

MINIMIZE T,
UNDER THE CONSTRAINTS

$$\begin{cases} (1a) & |\delta_i| \leq T \times b_i \\ (1b) & \delta_i \geq \alpha_i - T \times s_i \\ (1c) & \sum_i \delta_i = 0 \end{cases} \quad (1)$$

Fraction of load $f_{i,j}$

$$f_{i,j} = \delta_i \times \frac{\delta_j}{\sum_{k \in R} \delta_k} = \delta_i \times \frac{\delta_j}{-L}$$

Communication rate $\lambda_{i,j}$

$$\lambda_{i,j} = \frac{f_{i,j}}{T_0}$$

Computation rate $\gamma_{i,j}$

$$\gamma_{i,j} = \frac{f_{i,j}}{T_0}$$

Solution for Divisible Loads

Imbalance of a worker δ_i

Linear program

$$\begin{aligned}&\textsc{Minimize } T,\\&\textsc{under the constraints}\\&\left\{\begin{array}{ll}\text{(1a)} & |\delta_i| \leq T \times b_i\\ \text{(1b)} & \delta_i \geq \alpha_i - T \times s_i\\ \text{(1c)} & \sum_i \delta_i = 0\end{array}\right.\end{aligned} \quad (1)$$

Fraction of load $f_{i,j}$

$$f_{i,j} = \delta_i \times \frac{\delta_j}{\sum_{k \in R} \delta_k} = \delta_i \times \frac{\delta_j}{-L}$$

Communication rate $\lambda_{i,j}$

$$\lambda_{i,j} = \frac{f_{i,j}}{T_0}$$

Computation rate $\gamma_{i,j}$

$$\gamma_{i,j} = \frac{f_{i,j}}{T_0}$$

Outline

Conclusion

Complete study of a difficult load-balancing problem

Scheduling and redistributing data on master-slave platforms

Independent tasks:

- General case: Proof of NP-completeness in the strong sense
- Special platforms: Optimal algorithms
- Simulations: Verification of theoretical results

Divisible loads:

- Solution for general case: LP + analytical formulas

Perspectives

Beyond the NP-completeness: Search for approximation algorithms

Extension to dynamic master-slave platforms

Extension to more general interconnection networks

Conclusion

Complete study of a difficult load-balancing problem

Scheduling and redistributing data on master-slave platforms

Independent tasks:

- General case: Proof of NP-completeness in the strong sense
- Special platforms: Optimal algorithms
- Simulations: Verification of theoretical results

Divisible loads:

- Solution for general case: LP + analytical formulas

Perspectives

Beyond the NP-completeness: Search for approximation algorithms

Extension to dynamic master-slave platforms

Extension to more general interconnection networks

Recent work at our place

On-line scheduling heuristics for master-slave platforms

- Competitive ratios and inapproximability results
- Communication-aware heuristics

Collective communications

- Broadcast, multicast on heterogeneous clusters
- Resource selection for future MPI2 routines

Load-balancing

- Optimize BOINC-like applications
- Data redistribution strategies

Steady-state scheduling

- Multiple applications competing for resources
- Centralized vs fully distributed heuristics

Scheduling for large-scale platforms

Assess the impact of new architectural characteristics

- Heterogeneity
- Irregular network topologies
- Hierarchy
- Variability (volability)

Inject **static** knowledge in a (mostly) dynamic environment

- Divisible loads vs bag of tasks
- Steady-state scheduling
- Resource selection

Evaluation

- Evaluate strategies through simulation
- SimGrid software co-developed with UCSD
- Large-scale experiments with Grid'5000