DIET Scheduling for Ocean-Atmosphere Application

Yves Caniou, Eddy Caron, Ghislain Charrier, Andréea Chis, Frédéric Desprez, Éric Maisonnave, Vincent Pichon

LEGO

March 6, 2008
Outline

1. Ocean-Atmosphere
   - Description of the application
   - Scheduling for Ocean-Atmosphere Simulations

2. Adaptation to the grid
   - Repartition on the grid
   - Execution within DIET Simulations

3. Conclusion and Future work
Experimentation: several simulations (scenarios)
Scenarios: different initial parametrization
Scenario: chain of months
Input of the $n^{th}$ month = output of the $(n - 1)^{th}$ month
Chain of 2 consecutive monthly simulations.
First heuristic

- Disjoint groups of resources \( \min\{NS, \left\lceil \frac{R}{G} \right\}\) 
- Groups of the same size \(4 \rightarrow 11\) 
- Compute makespan for each possible grouping 
- Choose G yielding to the best makespan

Example of scheduling.
Weaknesses of this heuristic

- Resources can be idle during the whole execution
- Resources for post-processing can have idle time while waiting for new tasks to compute
**Improvements**

**Improvement 1.**
- Redistribute idle resources evenly among groups
- Resources for post-processing can still have idle time waiting for tasks to compute

**Improvement 2.**
- Redistribute all resources evenly among groups
- Maybe an unevenly distribution would be faster

**Improvement 3.**
- Model the problem as an instance of the Knapsack Problem
- Takes into account all previous defaults
Improvements

**Improvement 1.**
- Redistribute idle resources evenly among groups
- Resources for post-processing can still have idle time waiting for tasks to compute

**Improvement 2.**
- Redistribute all resources evenly among groups
- Maybe an unevenly distribution would be faster

**Improvement 3.**
- Model the problem as an instance of the Knapsack Problem
- Takes into account all previous defaults
Improvements

**Improvement 1.**
- Redistribute idle resources evenly among groups
- Resources for post-processing can still have idle time waiting for tasks to compute

**Improvement 2.**
- Redistribute all resources evenly among groups
- Maybe an unevenly distribution would be faster

**Improvement 3.**
- Model the problem as an instance of the Knapsack Problem
- Takes into account all previous defaults
Gains of each heuristic compared to the original one.
Outline

1 Ocean-Atmosphere
   - Description of the application
   - Scheduling for Ocean-Atmosphere
   - Simulations

2 Adaptation to the grid
   - Repartition on the grid
   - Execution within DIET
   - Simulations

3 Conclusion and Future work
Repartition on the grid

Description of the platform

- Several clusters
- Clusters are different from one another
- Clusters are homogeneous internally

Repartition on clusters

- Each cluster computes its completion time for 1 \( \rightarrow \) NS simulations
- Simulations are assigned one by one
- A simulation is assigned on the cluster where the augmentation of the makespan is the smallest
- Each cluster computes its assigned simulations
Repartition on the grid

Description of the platform
- Several clusters
- Clusters are different from one another
- Clusters are homogeneous internally

Repartition on clusters
- Each cluster computes its completion time for $1 \rightarrow NS$ simulations
- Simulations are assigned one by one
- A simulation is assigned on the cluster where the augmentation of the makespan is the smallest
- Each cluster computes its assigned simulations
Execution within a DIET architecture

**User**
- diet_async_call (as usual)

**API**
- Create profiles using estimation vector (6)
- Multiple submissions (7)
- Waits for all computations to be completed

**SeD**
Performance prediction (Makespan computation) (3)
Application execution (Scheduling) (8)

Repartition on several clusters.

Ghislain Charrier
DIET Scheduling for Ocean-Atmosphere Application
March 6, 2008
The client has to obtain the performances metrics. The client must be able to make several calls to clusters from one `diet_async_call()`.

Modification of the `diet_async_call` function, but not the API.
Simulations on several clusters.
Outline

1. Ocean-Atmosphere
   - Description of the application
   - Scheduling for Ocean-Atmosphere
   - Simulations

2. Adaptation to the grid
   - Repartition on the grid
   - Execution within DIET
   - Simulations

3. Conclusion and Future work
Conclusion

- Two cases: homogeneous and heterogeneous platforms
- Two level Scheduling
- The best heuristic uses the Knapsack modeling (linear constraints)
- The repartition of the work on the grid seems to give good results
Future work

- Continue integration within DIET (advanced stage of development)
- Make real experiments on Grid’5000
- For now, the multiple call is implemented directly in `diet_async_call()`. It has to be externalized so it can be used by other applications than Ocean-Atmosphere
- Make a proper implementation of the multiple call within DIET: need to add a new function in the API to obtain new unique IDs for each submission. This is necessary if we want vizdiet to work.
Thanks for your attention !!!