DE LA RECHERCHE À L'INDUSTRIE

Balance-Enforced Multi-Level Algorithm for Multi-Criteria Graph Partitioning

Rémi Barat [1,2]
Cédric Chevalier [1]
François Pellegrini[2,3]
[1] CEA, DAM, DIF, F-91297 Arpajon, France
[2] University of Bordeaux, France
[3] INRIA, France

www.cea.fr

12 October 2016
SIAM Combinatorial Scientific Computing

Outline

1. Objective
   - Context
   - Model
   - State of the art
2. Approach
   - The multi-level framework
   - Contributions
   - Example
3. Experiments
   - Mono-criterion partitioning (mesh of 3500 cells)
   - Multi-criteria partitioning (mesh of 3500 cells)
   - Multi-criteria partitioning (mesh of 22800 cells)

Outline

Context

High Performance Computing on distributed memory architectures.

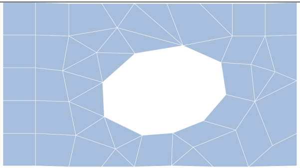
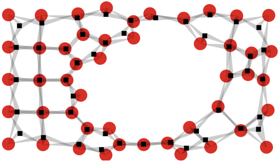
To get an efficient code, one must:

1. balance the workloads of each processor
2. overlap or minimize communications
3. take care of memory accesses
4. exploit full processor characteristics

We focus on the 1st and 2nd items.

Direct application: multi-physics numerical simulations using $2D$ or $3D$ meshes.

Hypergraph model

Mesh	Dual Hypergraph $H = (V, E)$
	
cell c_i	vertex $v_i \in V$
weight vector of a cell	weight vector of a vertex
c_i and its neighboring cells N_i	hyperedge $e = N_i \cup c_i \in E$
communicate c_i means y communications	weight y on the hyperedge corresponding to cell c_i

Problem : Hypergraph partitioning

Let p be the number of processors.
We search for an indexed family $(V_k)_{0 \leq k < p}$ of subsets of V pairwise disjoint and of union V, respecting:

1. some **constraints**: well-balanced workloads
2. an **objective**: minimize the communications.

NP-Hard Problem, no algorithm can always return the optimal solution.

State of the art

Main existing software:

Software	Representations	Multi-Criteria	Origin
Scotch	Topological	No	INRIA, F. Pellegrini et. al.
MeTiS	Topological	Yes	University of Minnesota, G. Karypis et. al.
Zoltan	Geometric Topological	Yes No	Sandia National Laboratories, K. Devine et. al.

Current limitations for the codes in CEA, DAM, DIF:

- Scotch does not fit: real need of a multi-criteria partitioner
- MeTiS does not meet the balance constraints
- Zoltan geometric representations are inefficient for our meshes

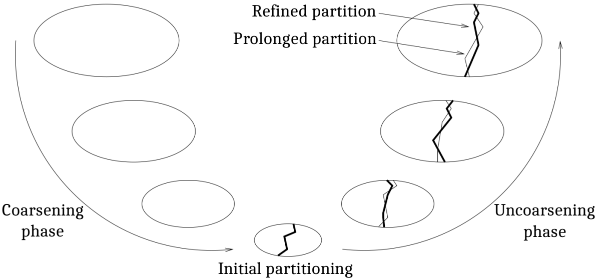
$\Rightarrow$ Lack of efficient multi-criteria partitioning tools.

Outline

Classic algorithm: The multi-level framework

A 3-phases algorithm:

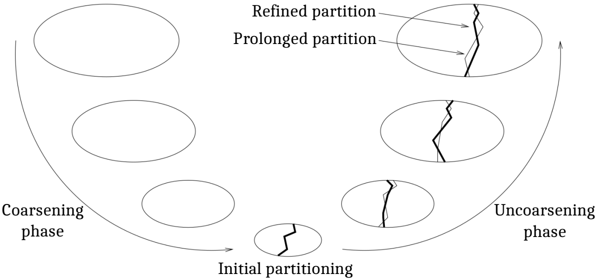
1. Coarsening
2. Initial partitioning of the coarsened hypergraph
3. Uncoarsening and refinement



Our approach: Multi-level multi-criteria algorithm

A 3-phases algorithm:

1. Coarsening
2. **Initial partitioning of the coarsened hypergraph**
   → New algorithm focusing on balance constraints
3. **Uncoarsening and refinement**
   → Adapted Fiduccia-Mattheyses algorithm



Initial partitioning algorithm

Problem: partition a set of vectors of numbers

- The vertices' weights alone are considered, not the hyperedges.
- Some algorithms exist in mono-criterion (number partitioning), but in our knowledge not in multi-criteria.

Algorithm 1 Initial partitioning algorithm

Require: V set of vertices, Π partition

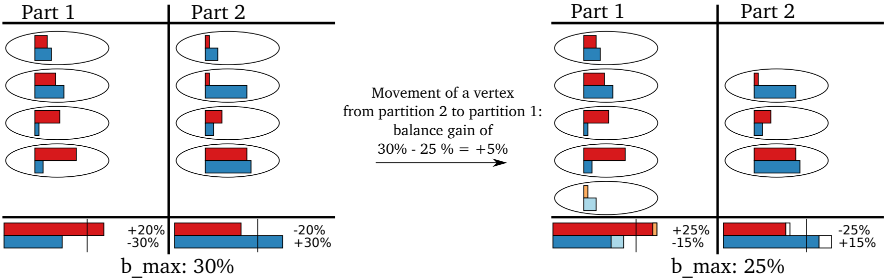
$b_{max} \leftarrow \max_{criterion\ c} \mathrm{Imbal}_c(\Pi)$
repeat
    for $v \in V$ **do**
        if changing partition of v decreases b_{max} **then**
            $\Pi \leftarrow$ change partition of v
            update b_{max}
        end if
    end for
until No more vertex move can decrease b_{max}

Example: initial partitioning

Simple instance:

- 8 vertices
- 2 criteria
- 2 partitions

Given a partition, choose a vertex to move:



Refinement algorithm: Fiduccia-Mattheyses

Key points:

- Move vertices according to their gain ("moves").
- Avoid opposite moves: lock on the moved vertices.
- When no more moves are possible: restore the best partition found.
- If improvement: start a new "pass". Otherwise, end of the algorithm.

Algorithm 2 Fiduccia-Mattheyses algorithm

```
Require: Partition respecting the constraints
  repeat                                                # Make a pass
    Unlock all vertices, compute their gains
    while possible moves remain do
      Move vertex of best gain and lock it
      Update neighbor gains and save current partition
    end while
    Restore the best partition reached in the pass
  until No improvement on the best partition quality
```

Refinement algorithm: Fiduccia-Mattheyses

Lots of possible variations:

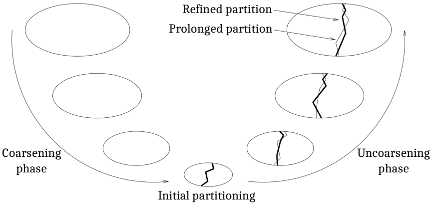
Options	Our choice	Scotch	MeTiS
Prescribed tolerance	strict	relaxed at lower levels	relaxed ($\propto \frac{1}{2 \times graph\ \ size}$)
Select move	best gain	best gain	best gain (if imbalanced: from the heaviest part for most imbalanced criterion)
Tie breaking	first	lowest imbalance	first
Inner loop stop condition (maximum number of moves of negative gain made in a row)	120		between 25 and 150 ($1\% \times graph\ \ size$)
Other remarks	hypergraph model	2 independent runs by default	rebalancing phases

Summary of the algorithm

Algorithmic contribution: multi-level for multi-criteria partitioning

1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

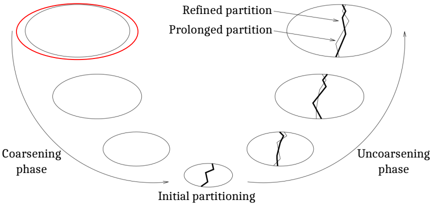
$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints



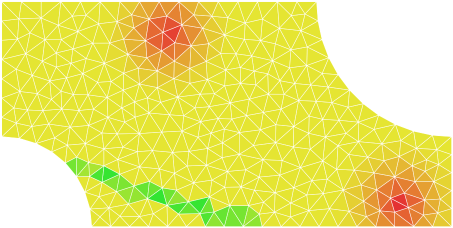
Algorithmic contribution: multi-level for multi-criteria partitioning

1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints



Mesh of 600 triangles
Vertex weights: 3 criteria
Edge weights depend on vertex weights

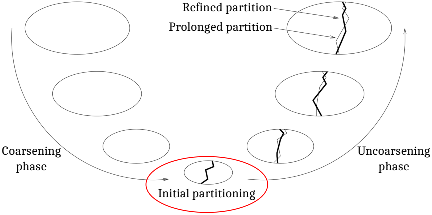


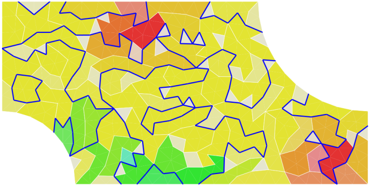
Summary of the algorithm
Example: initial partitioning

Algorithmic contribution: multi-level for multi-criteria partitioning

1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints





Initial partition of the coarsest hypergraph

imbalances	0.3%	2.7%	2.7%
communications		583	

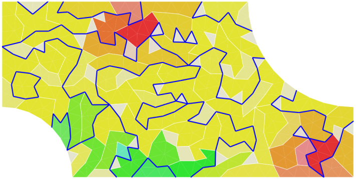
Summary of the algorithm

Example: uncoarsening and refinement

Algorithmic contribution: multi-level for multi-criteria partitioning

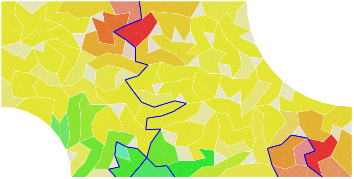
1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints



Initial partition (level -3)

imbalances	0.3%	2.7%	2.7%
communications	583		



Refinement (level -3)

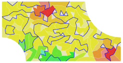
imbalances	0.2%	4.7%	4.8%
communications	197		

Summary of the algorithm
Example: uncoarsening and refinement

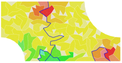
Algorithmic contribution: multi-level for multi-criteria partitioning

1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

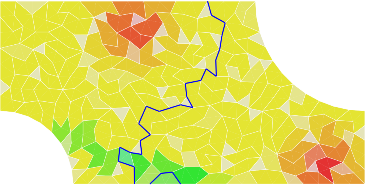
$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints



IP (-3)
coms: 583



R (-3)
coms: 197



Refinement (level -2)

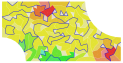
imbalances	2.3%	0.4%	4.7%
communications	97		

Summary of the algorithm
Example: uncoarsening and refinement

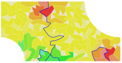
Algorithmic contribution: multi-level for multi-criteria partitioning

1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

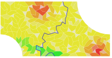
$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints



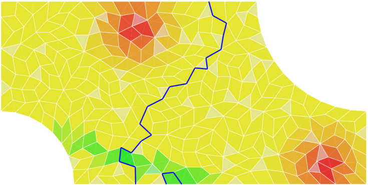
IP (-3)
coms: 583



R (-3)
coms: 197



R (-2)
coms: 97



Refinement (level -1)

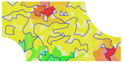
imbalances	2.6%	0.6%	2.0%
communications		52	

Summary of the algorithm
Example: uncoarsening and refinement

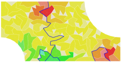
Algorithmic contribution: multi-level for multi-criteria partitioning

1. Classic coarsening (Heavy-Edge Matching)
2. Greedy initial partitioning returning a solution respecting the balance constraints
3. Refinement of the objective function respecting the balance constraints

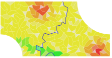
$\Longrightarrow$ Each solution found is **guaranteed** to respect all balance constraints



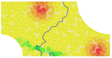
IP (-3)
coms: 583



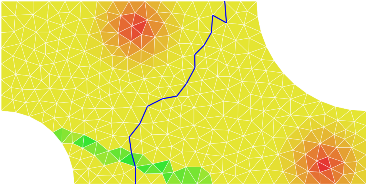
R (-3)
coms: 197



R (-2)
coms: 97



R (-1)
coms: 52



Final partition (level 0)

imbalances	0.8%	0.6%	0.8%
communications		28	

Outline

1. Objective
   - Context
   - Model
   - State of the art
2. Approach
   - The multi-level framework
   - Contributions
   - Example
3. Experiments
   - Mono-criterion partitioning (mesh of 3500 cells)
   - Multi-criteria partitioning (mesh of 3500 cells)
   - Multi-criteria partitioning (mesh of 22800 cells)

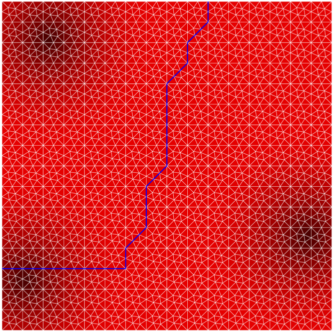
Experiment 1
Comparison with MeTiS and Scotch (mono-criterion)

Instance

# cells	3500
vertex weights statistics:	
min	10
max	2457
average	318
std	507
edge weights:	
hypergraph model	weight of cell
graph model	sum of weights of ends

Parameters

runs	500 (random numbering of the graph vertices for each run)
tolerance	5%
MeTiS version	5.1.0[1]
Scotch version	6.0.4[2]



Bi-partition example
The darker a cell, the heavier its weight
Blue line: border

[1] MeTiS is used with vertex sizes provided, so that it minimizes exactly communication volume (unlike Scotch which minimizes the edge-cut).

[2] By default, Scotch launches 2 independent runs and returns the best partition found.

Experiment 1
Comparison with MeTiS and Scotch (mono-criterion)

Software	Our algorithm	MeTiS	Scotch
constraints			
valid solutions	100%	100%	100%
communications			
average	3756	5392	3519
std	1047	751	535
min	2431	2908	2443
median	3434	5482	3514
max	8551	6959	5301

Observations:

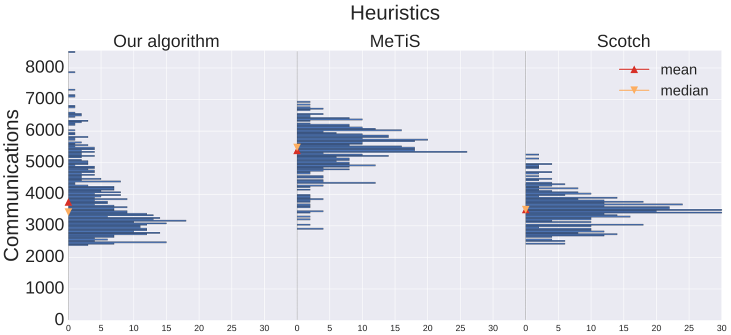
- Scotch is the best
- Our algorithm statistics seem close

Experiment 1
Comparison with MeTiS and Scotch (mono-criterion)

Software	Our algorithm	MeTiS	Scotch
constraints			
valid solutions	100%	100%	100%
communications			
average	3756	5392	3519
std	1047	751	535
min	2431	2908	2443
median	3434	5482	3514
max	8551	6959	5301

Observations:

- Very different behaviors
- High discrepancy



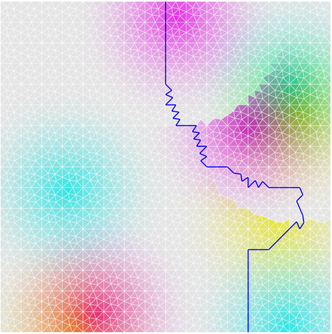
Experiment 2
Comparison with MeTiS (multi-criteria)

Instance

# cells			3500
vertex weights statistics (3 criteria):			
min	10	10	10
max	2487	2403	2464
average	296	288	257
std	473	448	444
edge weights:			
hypergraph model			1st weight of cell
graph model			sum of 1st weights of ends

Parameters

runs	500 (random numbering of the graph vertices for each run)
tolerance	5%
MeTiS version	5.1.0[1]



Bi-partition example
One color = one criterion
Blue line: border

[1] MeTiS is used with vertex sizes provided.

Experiment 2
Comparison with MeTiS (multi-criteria)

Software	Our algorithm	MeTiS
constraints statistics:		
valid solutions	100%	60%
communication statistics:		
average	2733	2436
std	2316	1729
min	215	340
median	1888	1839
max	9673	6093

Observations:

- MeTiS seems to achieve better performance in terms of partition quality
- However, its policy to relax constraints leads to invalid solutions

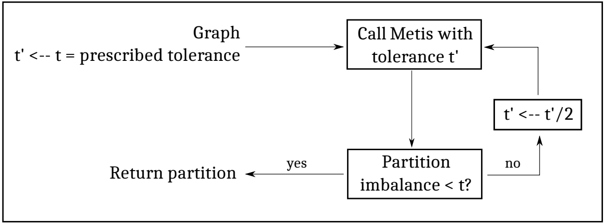
Experiment 2
Comparison with MeTiS (multi-criteria)

Software	Our algorithm	MeTiS	Failsafe-MeTiS
constraints statistics:			
valid solutions	100%	60%	100%
communication statistics:			
average	2733	2436	
std	2316	1729	
min	215	340	
median	1888	1839	
max	9673	6093	

Observations:

- Failsafe-MeTiS: if solution found is invalid, relaunched with half-tolerance.

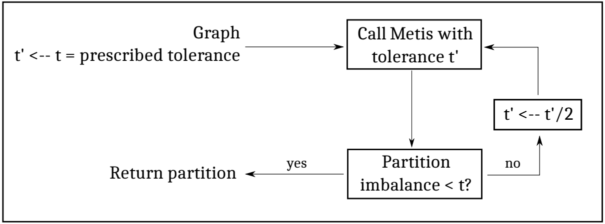
Failsafe-MeTiS



Experiment 2
Comparison with MeTiS (multi-criteria)

Software	Our algorithm	MeTiS	Failsafe-MeTiS
constraints statistics:			
valid solutions	100%	60%	100%
communication statistics:			
average	2733	2436	2291
std	2316	1729	1517
min	215	340	340
median	1888	1839	1787
max	9673	6093	6093

Failsafe-MeTiS

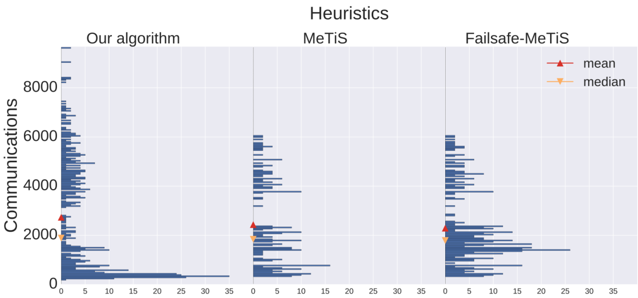


Observations:

- Failsafe-MeTiS: if solution found is invalid, relaunched with half-tolerance.
- Better performance when constraints are tougher!

Experiment 2
Comparison with MeTiS (multi-criteria)

Software	Our algorithm	MeTiS	Failsafe-MeTiS
constraints statistics:			
valid solutions	100%	60%	100%
communication statistics:			
average	2733	2436	2291
std	2316	1729	1517
min	215	340	340
median	1888	1839	1787
max	9673	6093	6093



Observations:

- The comparison is less straightforward
- Our algorithm gets lots of solutions of very good quality
- ...but also some of very bad quality
- Relaxing the constraints does not lead to better solutions more often here
- The discrepancy is greater for this instance.

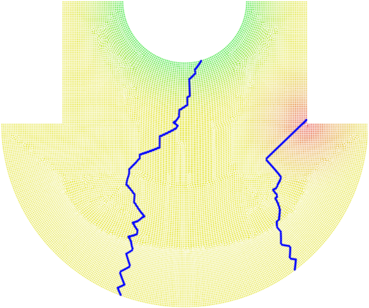
Experiment 3
Comparison with MeTiS (multi-criteria)

Instance

# cells			22800
vertex weights statistics (3 criteria):			
min	10	10	1
max	2403	9671	1
average	148	322	1
std	418	1074	0
edge weights:			
hypergraph model			1st weight of cell
graph model			sum of 1st weights of ends

Parameters

runs	60 (random numbering of the graph vertices for each run)
tolerance	5%
MeTiS version	5.1.0[1]

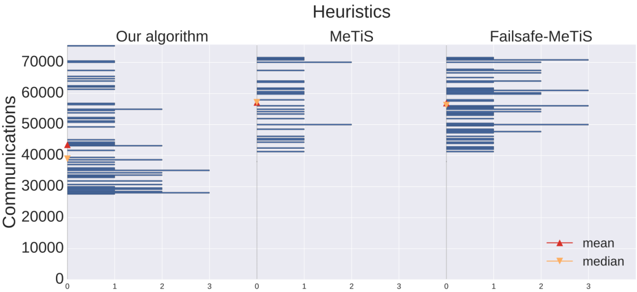


Bi-partition example
One color = one criterion
Blue line: border

[1] MeTiS is used with vertex sizes provided.

Experiment 3
Comparison with MeTiS (multi-criteria)

Software	Our algorithm	MeTiS	Failsafe-MeTiS
runs	60	60	60
constraints statistics:			
valid solutions	100%	47%	100%
communication statistics: $(\times 1000)$			
average	43.4	57.1	56.8
std	13.5	9.5	8.8
min	28.0	41.5	41.5
median	38.9	57.1	56.2
max	75.7	71.6	71.6



Observations:

- MeTiS returns lots of invalid solutions, but does not perform better than Failsafe-MeTiS.
- Our algorithm reaches better partitions for this instance.
- Still a very high discrepancy, no matter the tool.

- Objective : accelerate multi-physics simulations by balancing the workload and minimizing the communications
- Approach and contributions:
  - Adaptation of the multi-level framework to multi-criteria graphs or hypergraphs
  - New initial partitioning algorithm
  - Refinement respecting the balance constraints
- Implementation of a Python prototype
- Comparison with some existing tools:
  - Studies more precisely the algorithms behavior
  - Shows their lack of robustness
  - Questions MeTiS policy to relax constraints

Conclusion
Perspectives

- *Currently:* implementation (open-source) of the multi-criteria algorithms in Scotch
  $\Longrightarrow$ Validation on real size instances
  $\Longrightarrow$ Validation on a simulation code
  $\Longrightarrow$ New release next year
- Enforce the algorithm robustness by:
  - Analyzing the algorithms behavior
  - Studying the influence of each parameter
  - Working on the graph numbering
- Set up of a parallel version of the algorithms

Thank you

Commissariat à l'énergie atomique et aux énergies alternatives
Centre DAM Île-de-France | F-91297 Arpajon
T. +33 (0)1 69 26 40 00