Energy-aware mappings of series-parallel workflows onto chip multiprocessors

Anne Benoit [†], Rami Melhem[‡],
Paul Renaud-Goud [†] and Yves Robert [†]

[†] École Normale Supérieure de Lyon,
France

[‡] University of Pittsburgh,
USA

Scheduling in Aussois

Motivation and introduction

- Energy saving: crucial problem
- Chip MultiProcessor (CMP): present and future of the processor
- Streaming applications: ubiquitous in many domains, like image processing
- Most task graphs are Series-Parallel Graphs (SPGs)

Outline of the talk

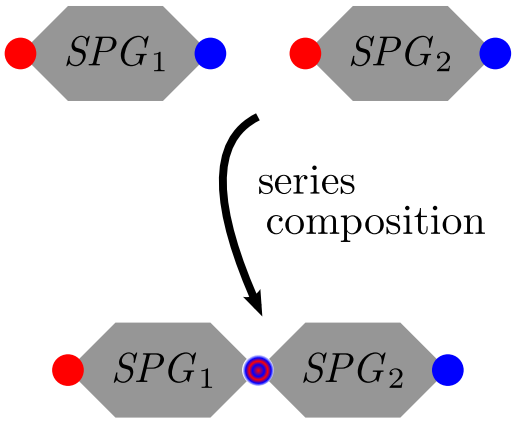
Outline of the talk

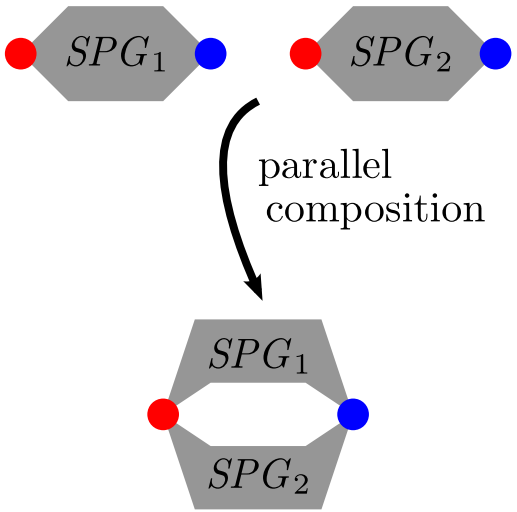
Series-parallel graph

- G is a SPG if G is a composition of two SPGs
-  is a SPG
- Compositions

Series-parallel graph

- G is a SPG if G is a composition of two SPGs
-  is a SPG
- Compositions





Series-parallel graph

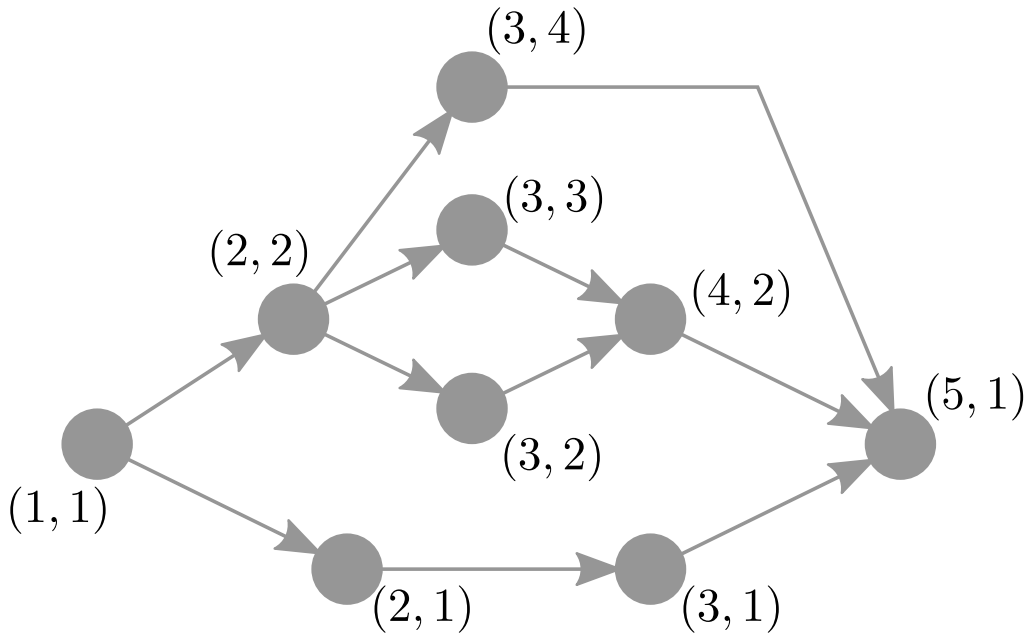
- G is a SPG if G is a composition of two SPGs
-  is a SPG
- Compositions: series, parallel
- Label

Series-parallel graph

- G is a SPG if G is a composition of two SPGs
-  is a SPG
- Compositions: series, parallel
- Label

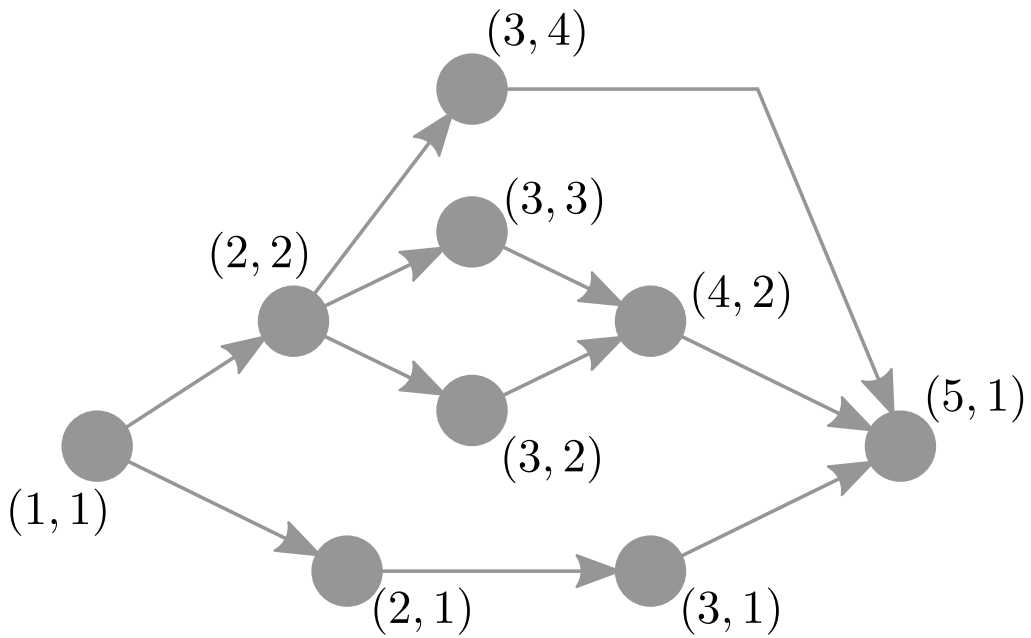
Series-parallel graph

- G is a SPG if G is a composition of two SPGs
-  is a SPG
- Compositions: series, parallel
- Label



Series-parallel graph

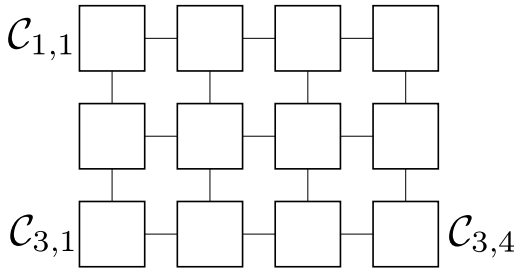
- G is a SPG if G is a composition of two SPGs
- ●—● is a SPG
- Compositions: series, parallel
- Label



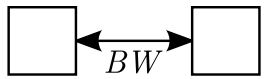
$\Rightarrow$ bounded elevation

Target platform

- Cores $\mathcal{C}_{u,v}$ on a $p \times q$ grid

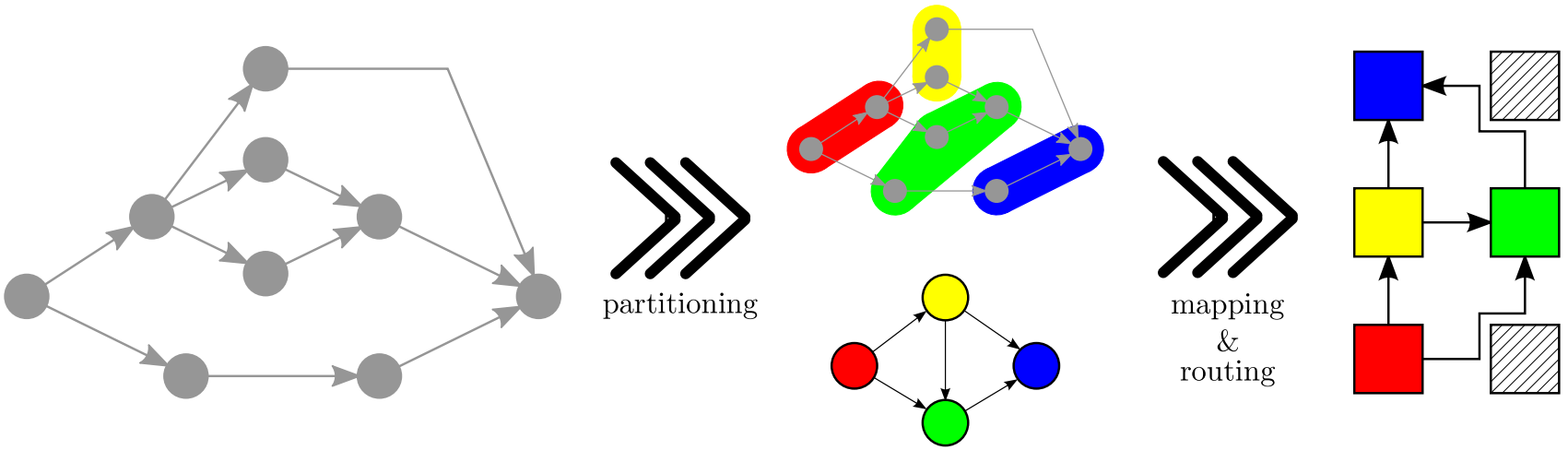


- Bidirectional links of bandwidth BW



- $\mathcal{C}_{u,v}$ running at speed $s_{u,v} \in \{s^{(1)}, \ldots, s^{(M)}\}$

Mapping strategy

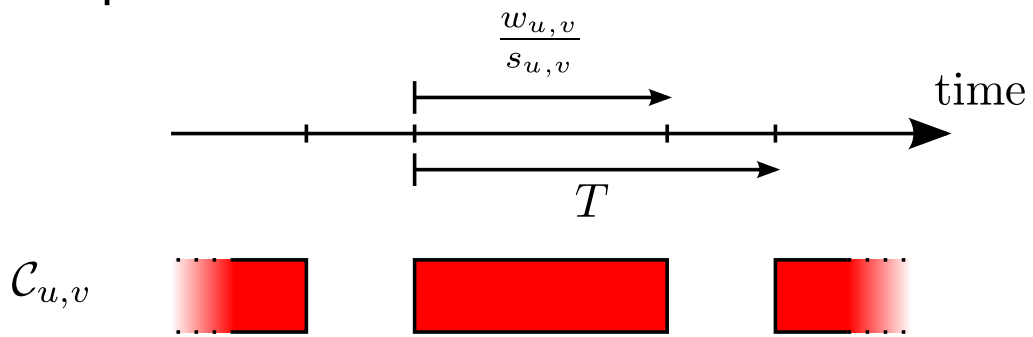


- Partitioning: if S_i and S_j mapped onto $\mathcal{C}_{u,v}$ and $S_i \rightarrow S_k \rightarrow S_j$, then S_k mapped onto $\mathcal{C}_{u,v}$
- Mapping:
  - One-to-one
  - Routing free, but single-path

$\Rightarrow$ $w_{u,v}$: work computed by $\mathcal{C}_{u,v}$
$b_{(u,v)\rightarrow(u',v')}$: communication from $\mathcal{C}_{u,v}$ to $\mathcal{C}_{u',v'}$

Objective functions

- Given period T
- Energy of computations



$$E^{(\text{comp})} = |\mathcal{A}| \times P_{\text{leak}}^{(\text{comp})} \times T + \sum_{\mathcal{C}_{u,v} \in \mathcal{A}} \frac{w_{u,v}}{s_{u,v}} \times P_{s_{u,v}}^{(\text{comp})},$$

where $\mathcal{A}$ is the set of active cores

- Energy of communications

$$E^{(\text{comm})} = P_{\text{leak}}^{(\text{comm})} \times T + \left(\sum_{u,v} \sum_{u',v'} b_{(u,v)\to(u',v')} \right) \times E^{(\text{bit})}$$

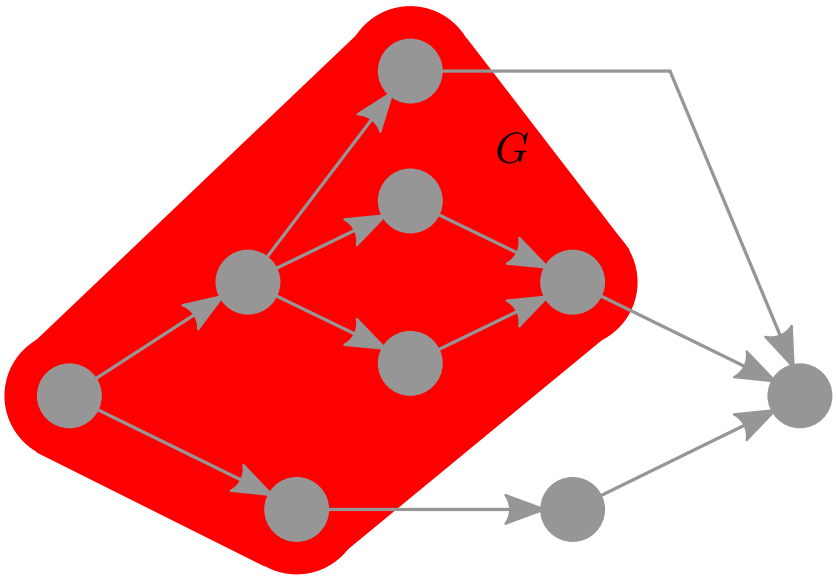
Outline of the talk

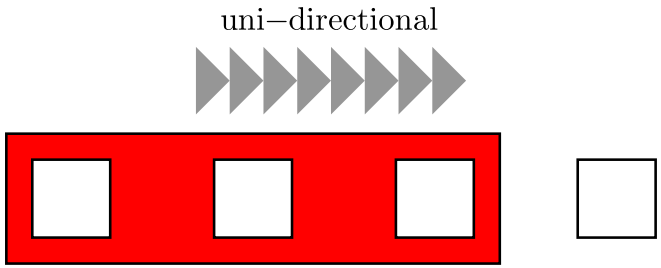
Uni-directional uni-line CMP

Polynomial with bounded elevation:
dynamic programming algorithm

$$\mathcal{E}(G,k) = \min_{G' \subseteq G} \left(\mathcal{E}(G', k-1) + \mathcal{E}^{\text{cal}}(G \setminus G') \right) ,$$

where
- G' is admissible
- outgoing communications of G' do not exceed BW



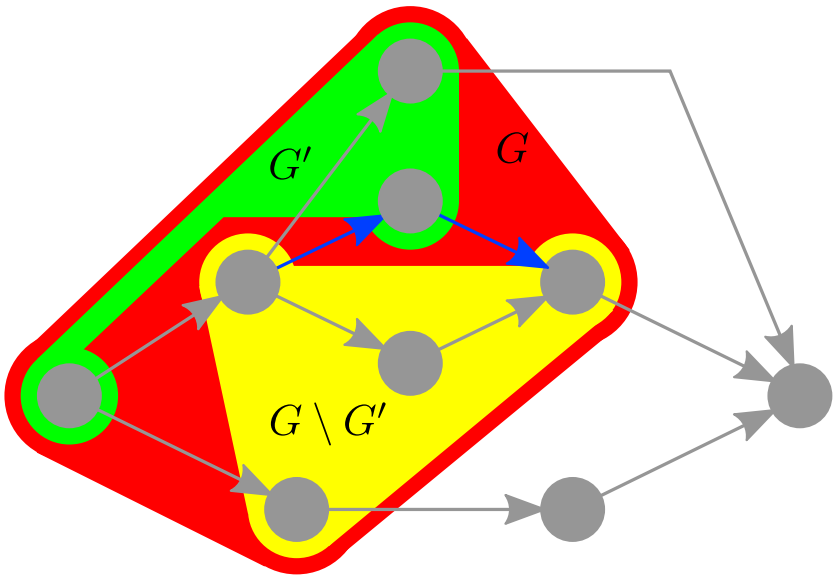


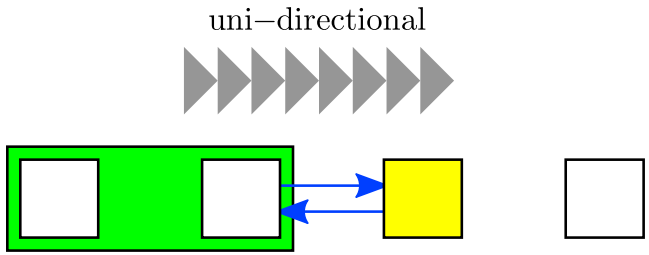
Uni-directional uni-line CMP

Polynomial with bounded elevation:
dynamic programming algorithm

$$\mathcal{E}(G,k) = \min_{G' \subseteq G} \left(\mathcal{E}(G', k-1) + \mathcal{E}^{\text{cal}}(G \setminus G') \right) ,$$

where
- G' is admissible
- outgoing communications of G' do not exceed BW



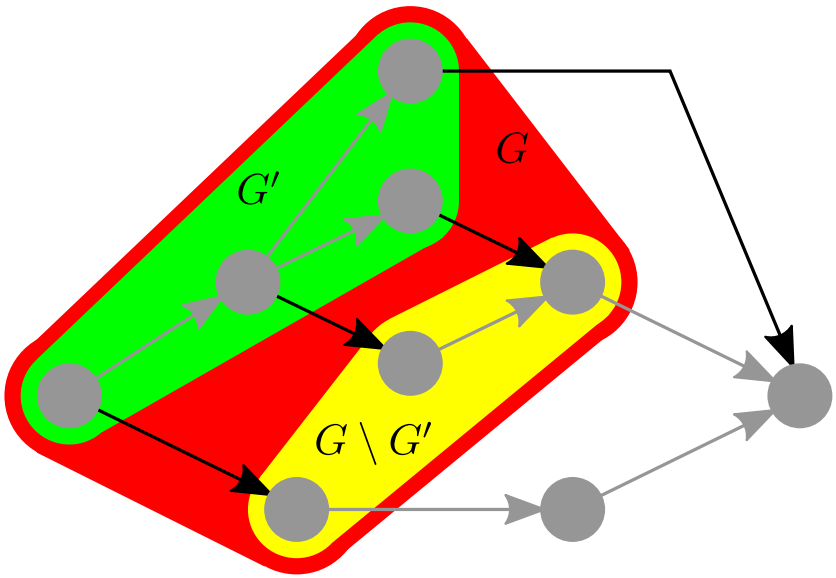


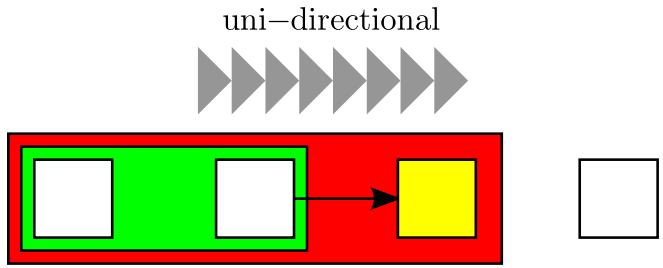
Uni-directional uni-line CMP

Polynomial with bounded elevation:
dynamic programming algorithm

$$\mathcal{E}(G,k) = \min_{G' \subseteq G} \left(\mathcal{E}(G', k-1) + \mathcal{E}^{\text{cal}}(G \setminus G') \right) ,$$

where
- G' is admissible
- outgoing communications of G' do not exceed BW



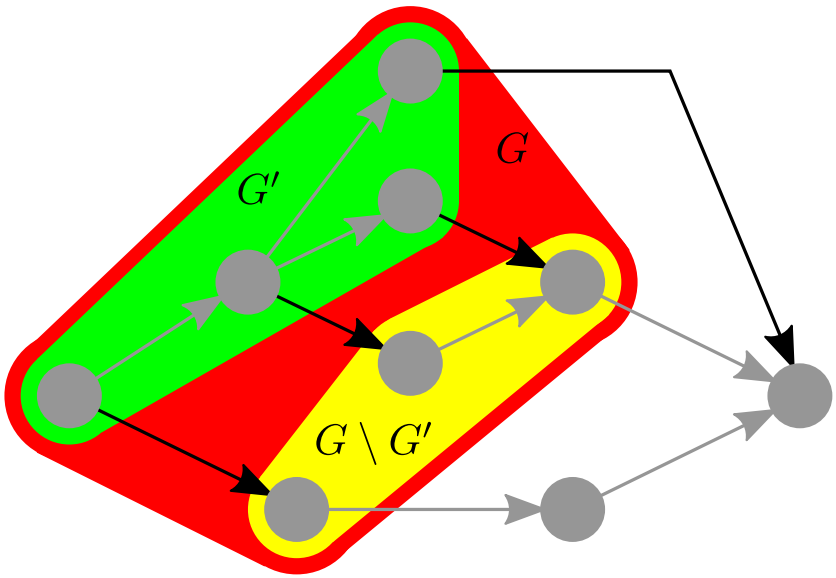


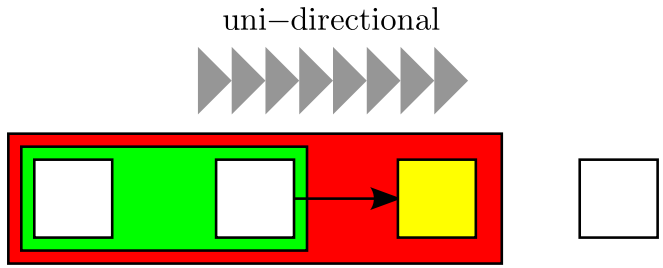
Uni-directional uni-line CMP

Polynomial with bounded elevation:
dynamic programming algorithm

$$\mathcal{E}(G,k) = \min_{G' \subseteq G} \left(\mathcal{E}(G', k-1) + \mathcal{E}^{\text{cal}}(G \setminus G') \right) ,$$

where
- G' is admissible
- outgoing communications of G' do not exceed BW

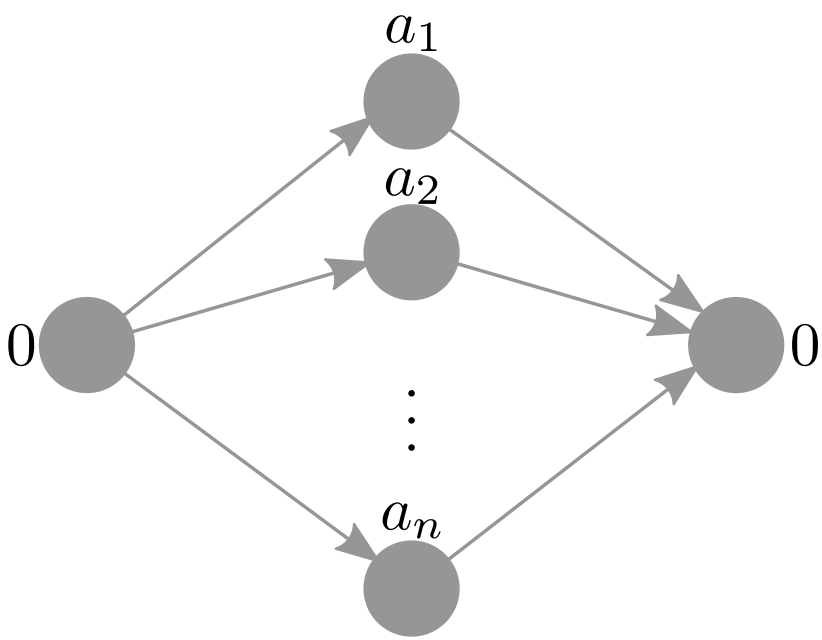


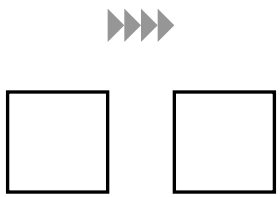


Polynomial
$O(q \times n^{y_{\max}+1})$

Uni-directional uni-line CMP

NP-complete with unbounded elevation:
reduction from 2-PARTITION

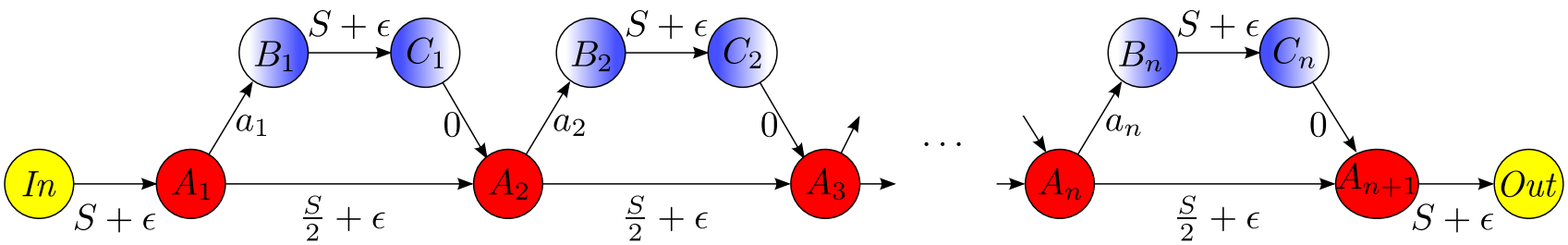




Single speed $\frac{\sum_{i=1}^{n} a_i}{2}$

Bi-directional uni-line CMP

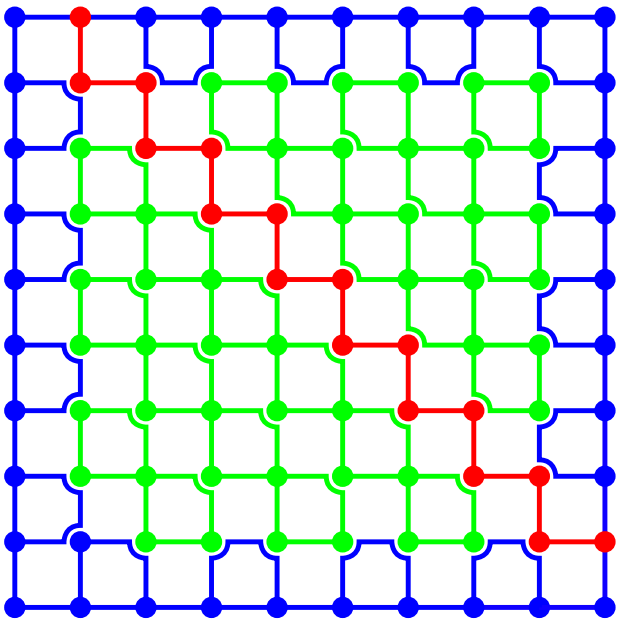
NP-complete with bounded elevation:
reduction from 2-PARTITION

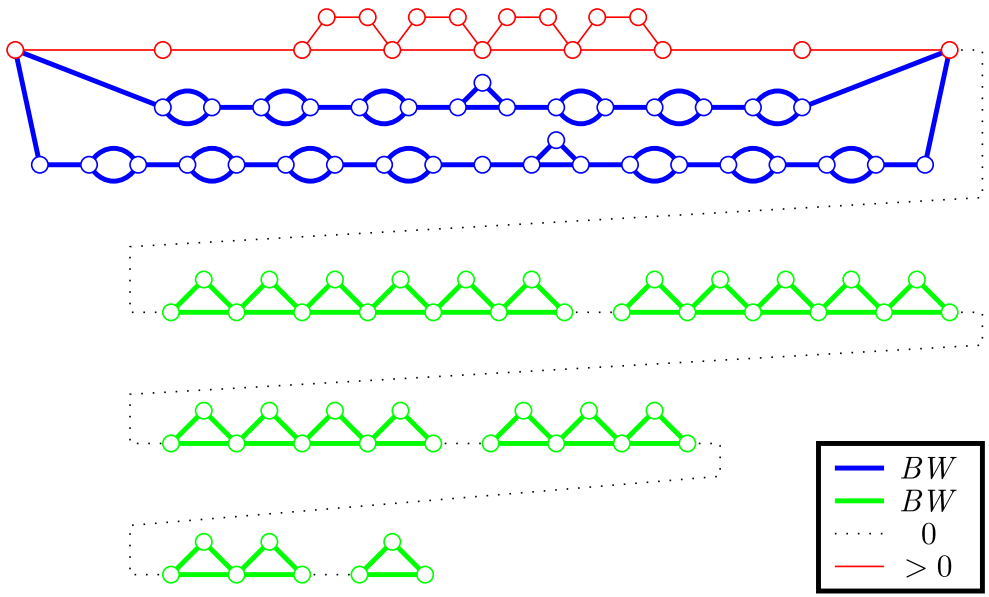


$$BW = \frac{3S}{2} + \epsilon$$

Bi-directional square CMP

NP-complete with bounded elevation:
reutilization of previous proof



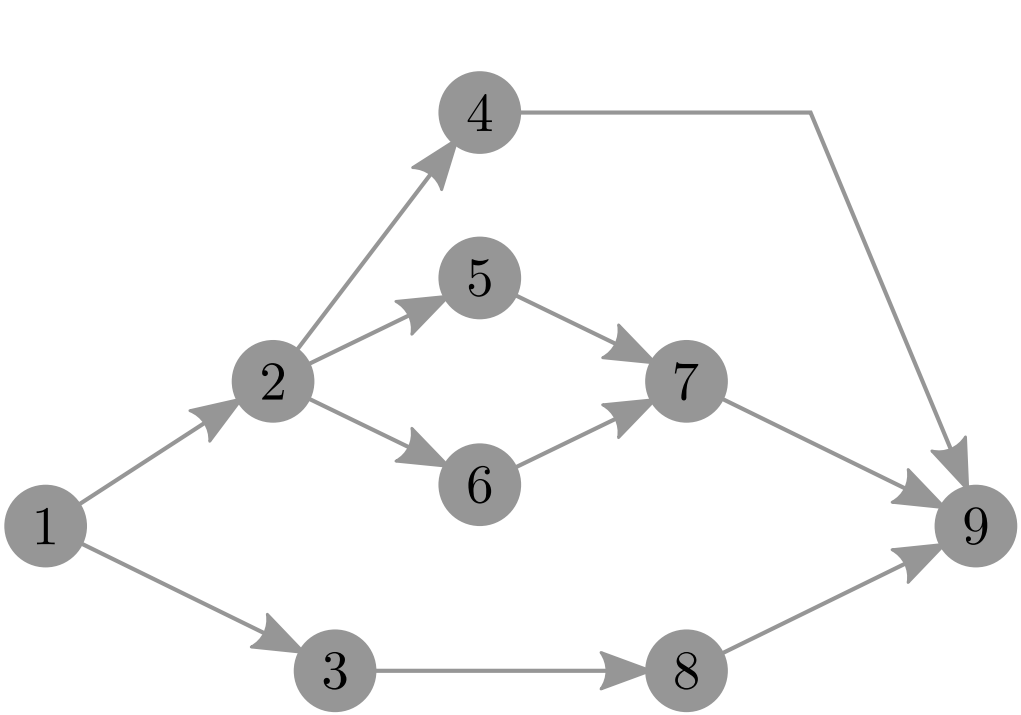


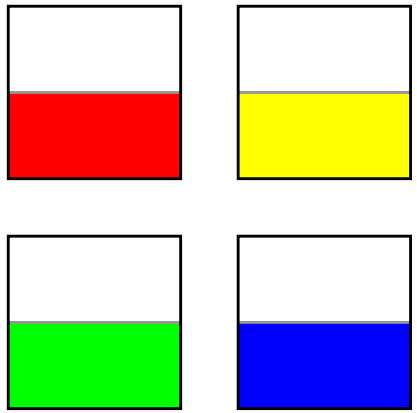
Outline of the talk

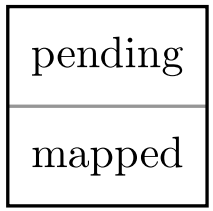
Heuristic summary

- Random heuristic: random speeds, random assignments, XY routing
- Greedy heuristic: starting from $\mathcal{C}_{1,1}$, process as many nodes as possible, give following nodes to right and down cores; iterate on those cores
- 2D dynamic programming algorithm
- 1D heuristics (2D CMP configured as a snake):
  - Optimal solution on 1D uni-directional CMP
  - Previous 2D dynamic programming algorithm onto snake

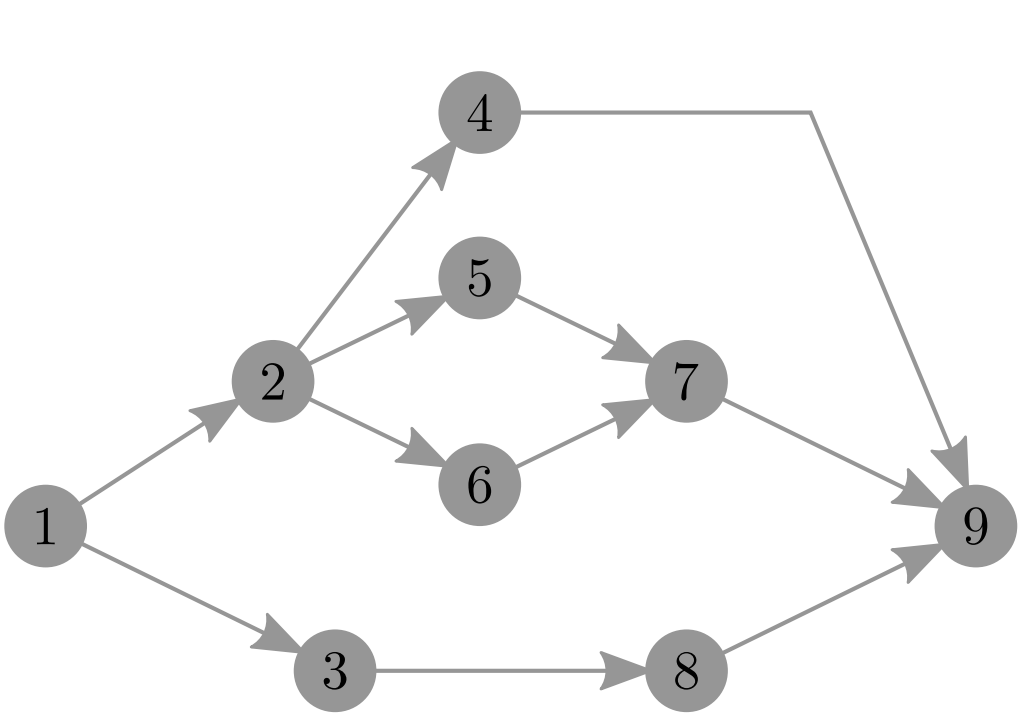
Greedy

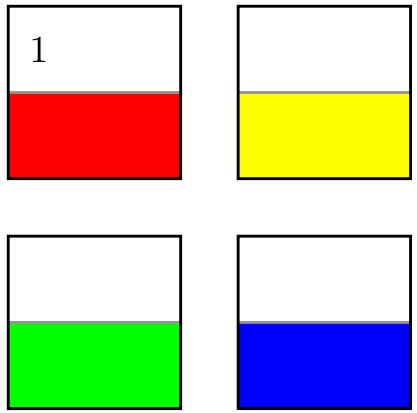


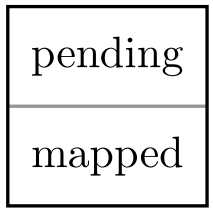




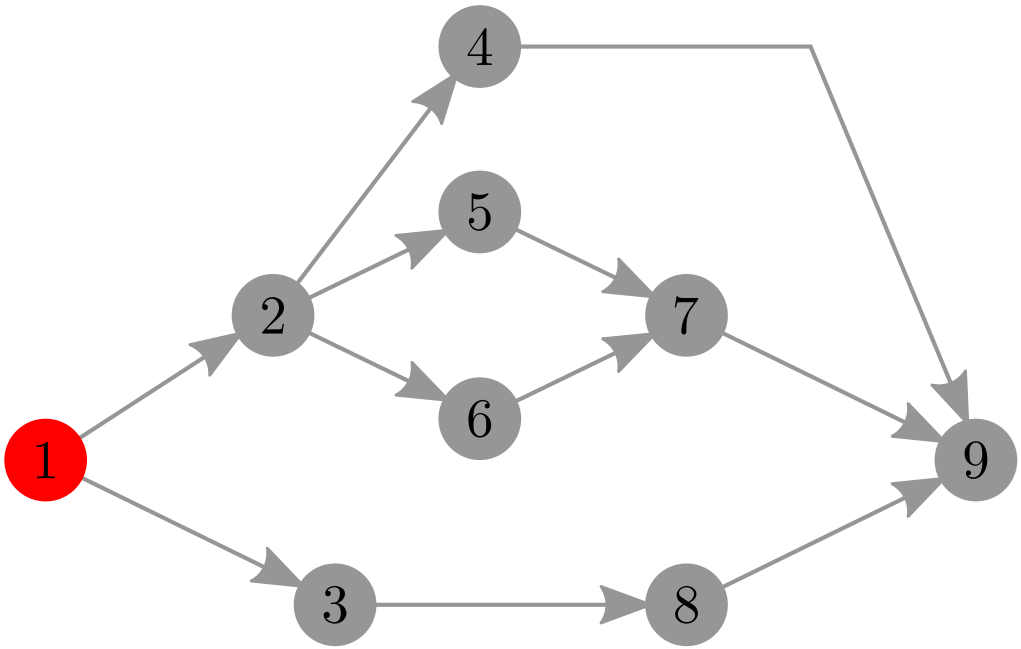
Greedy

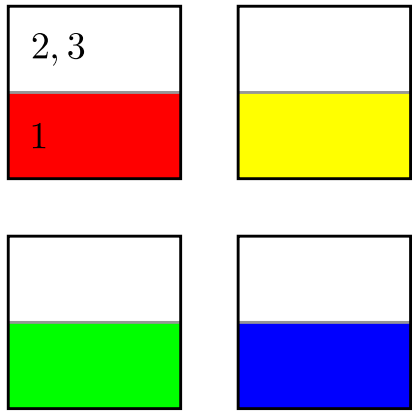


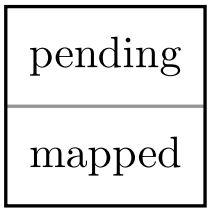




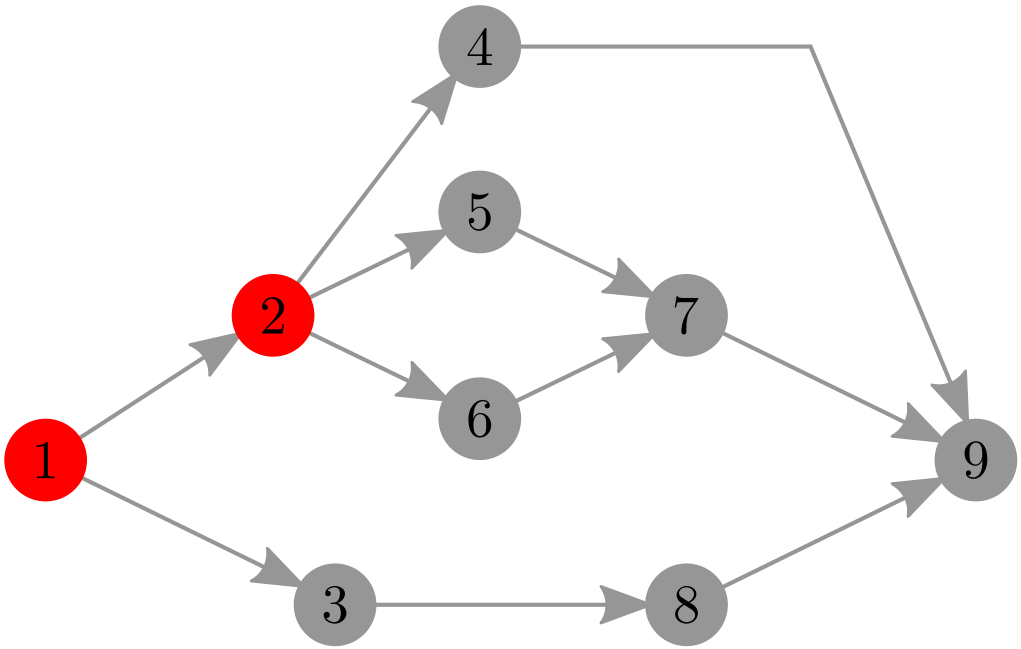
Greedy

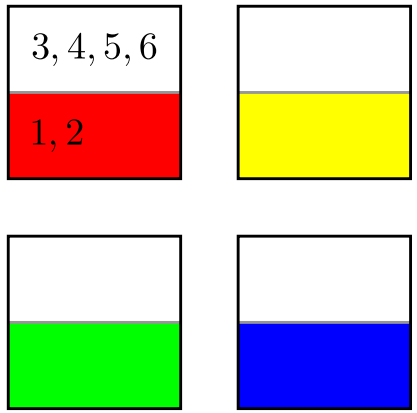


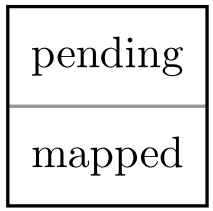




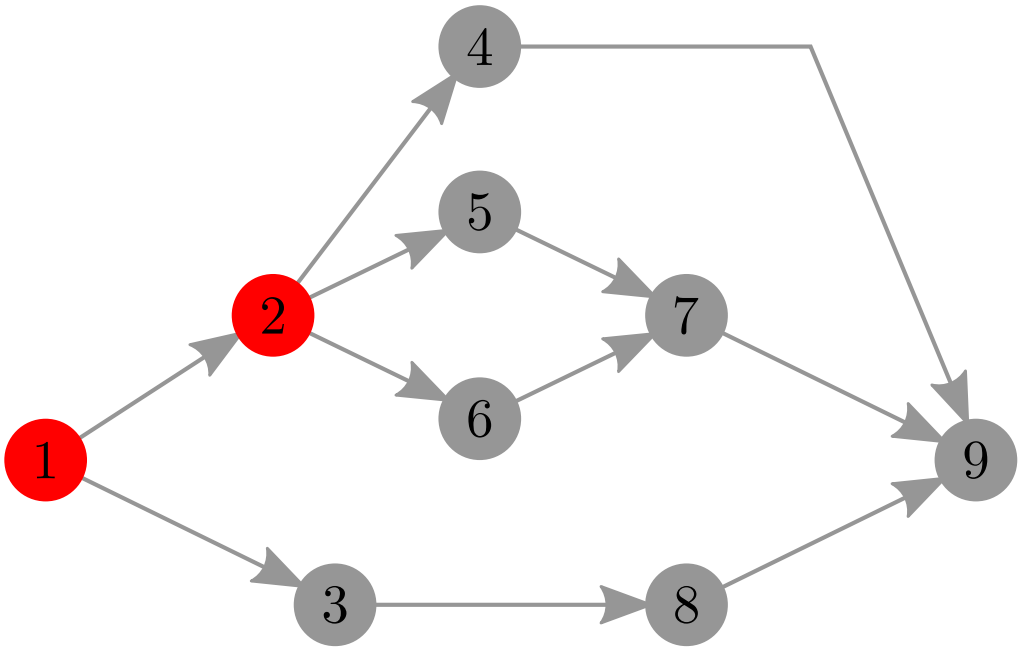
Greedy

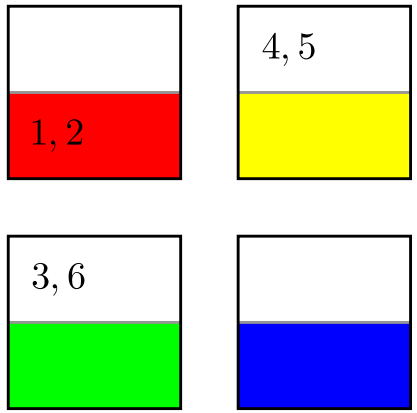


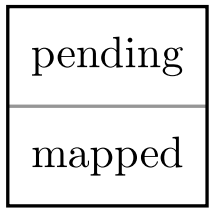




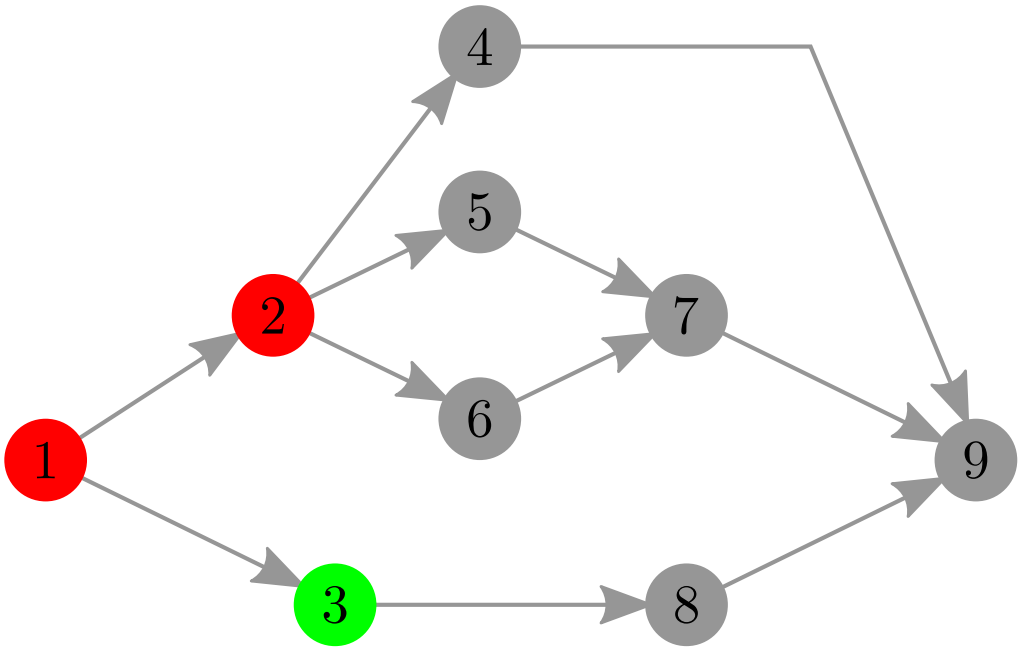
Greedy

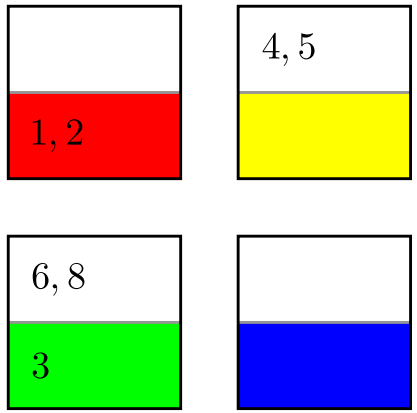


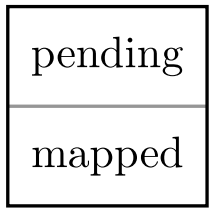




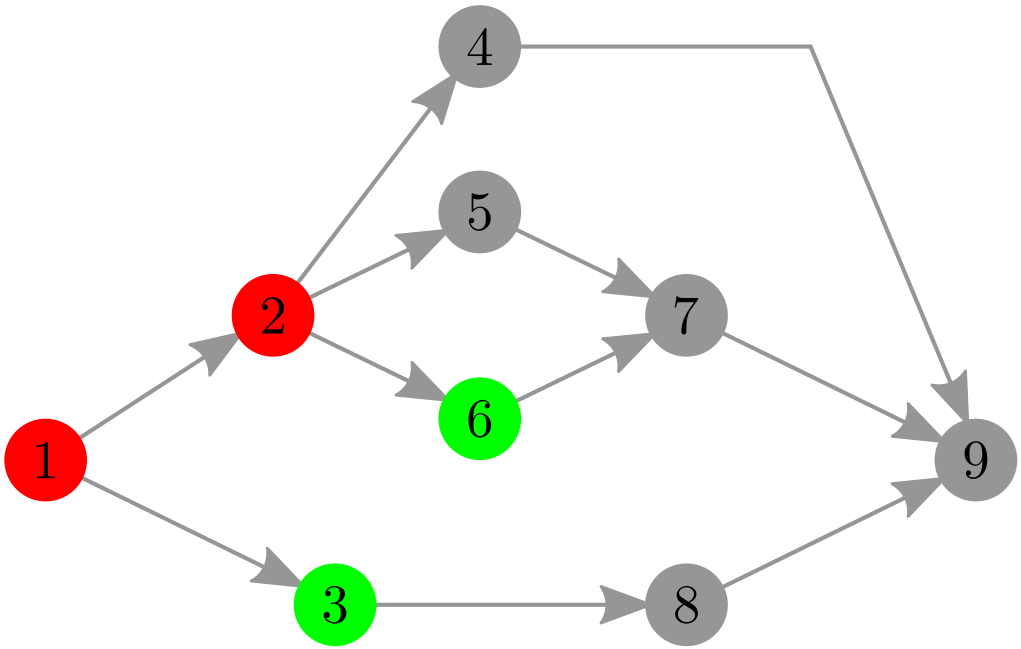
Greedy

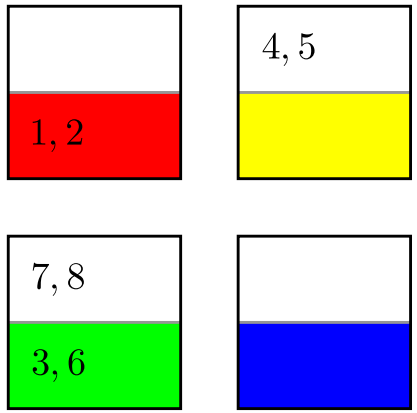


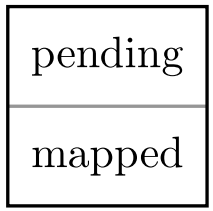




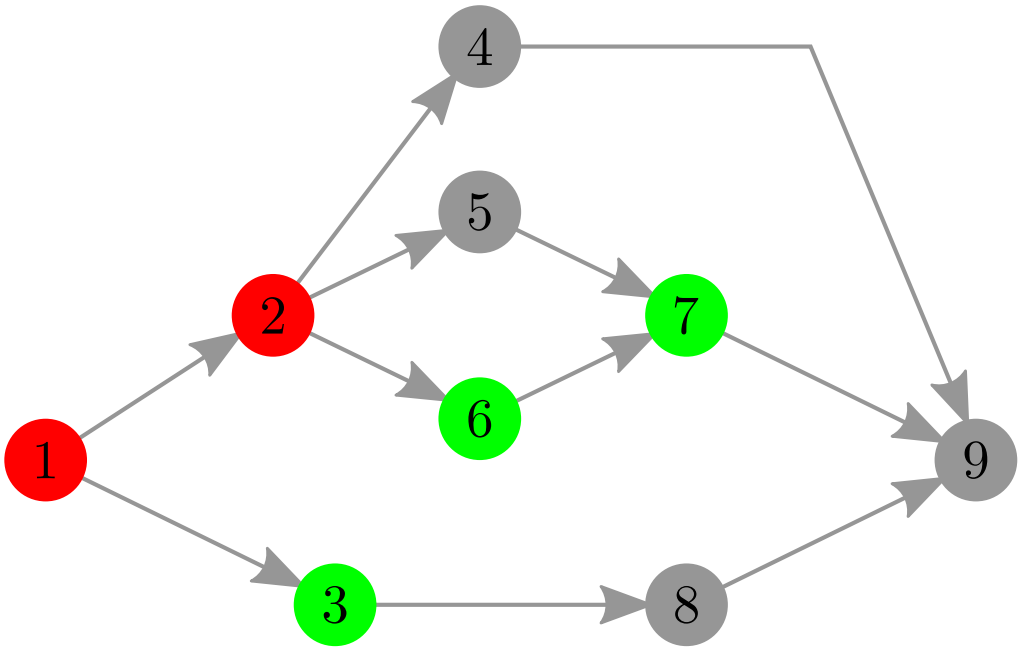
Greedy

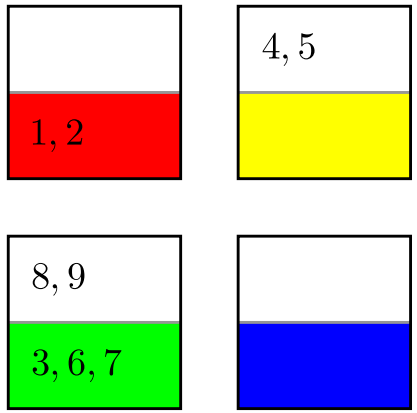


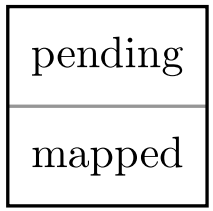




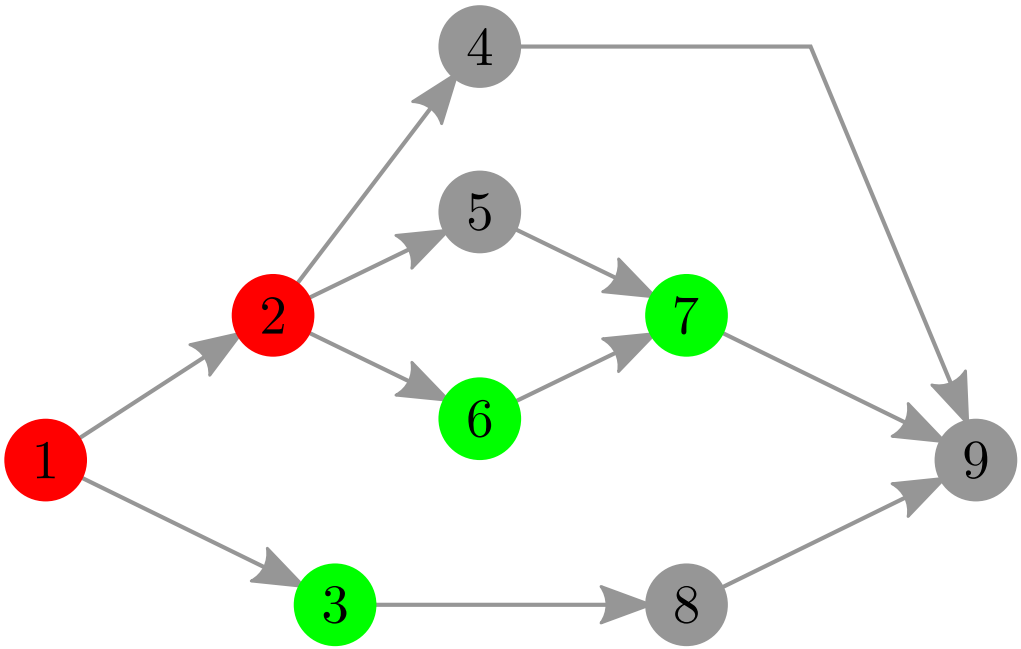
Greedy

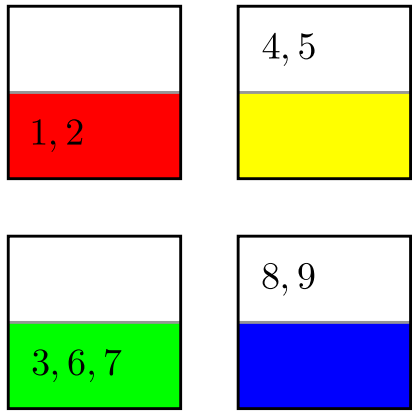


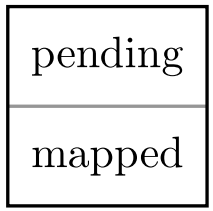




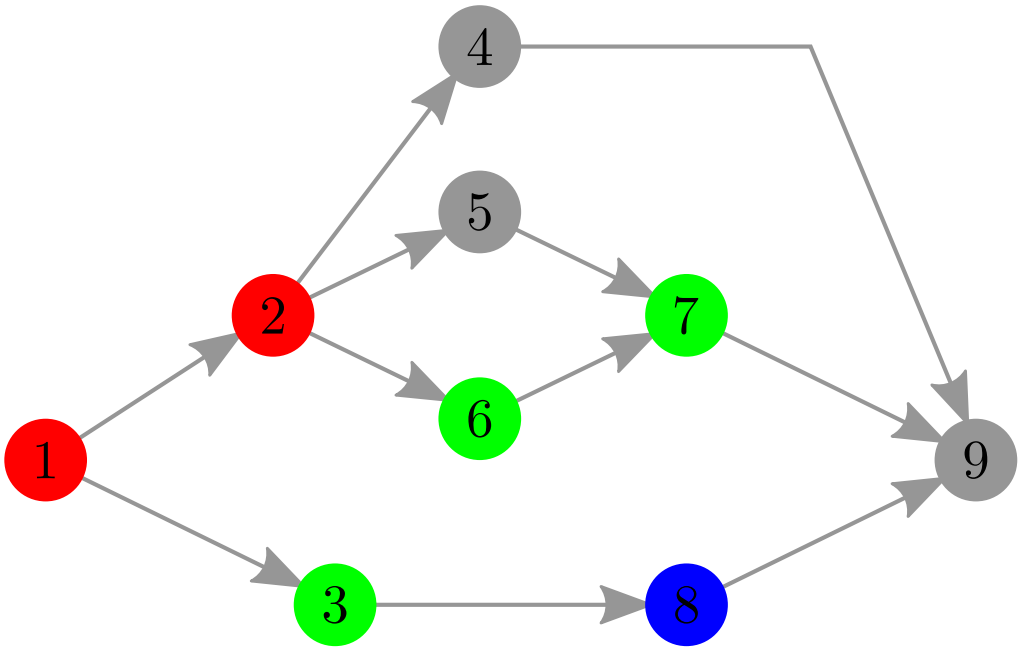
Greedy

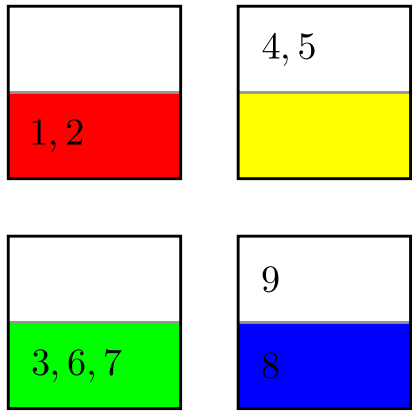


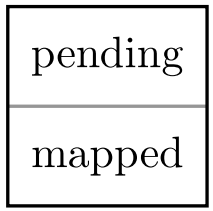




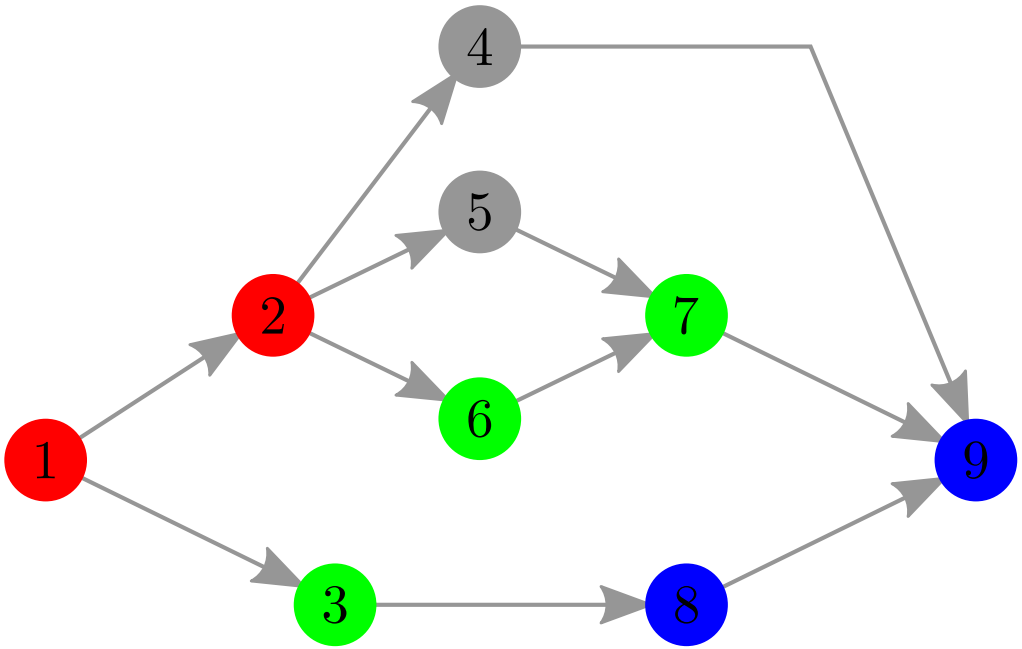
Greedy

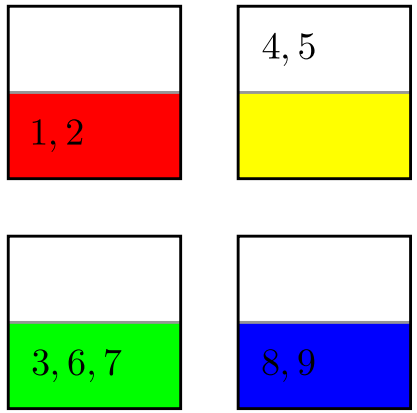


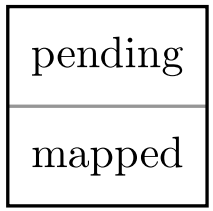




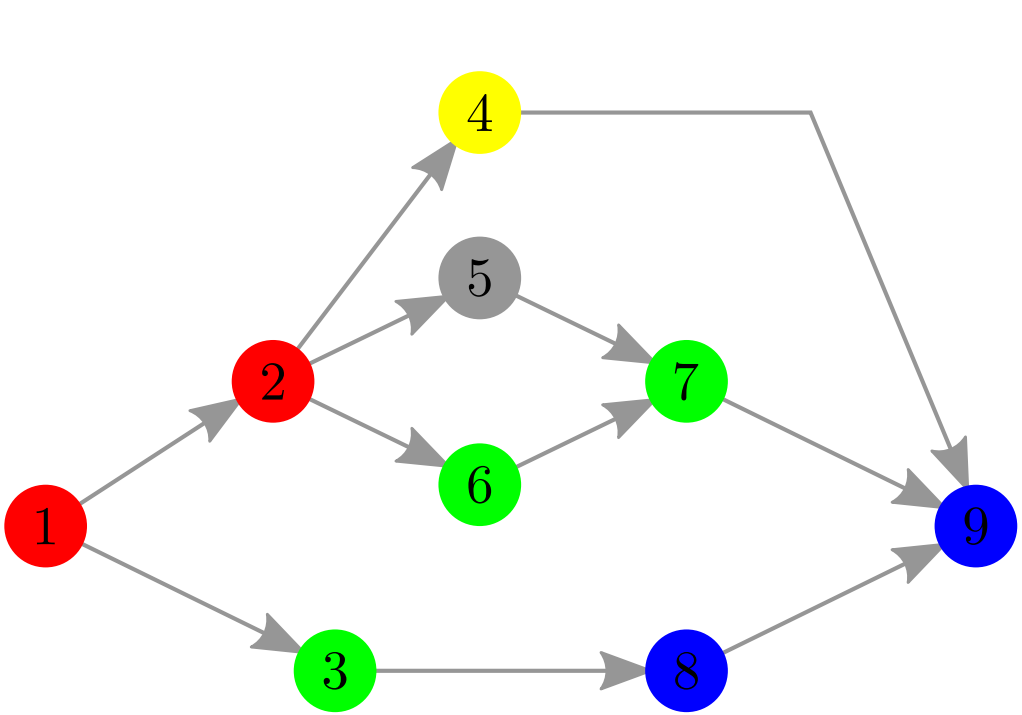
Greedy

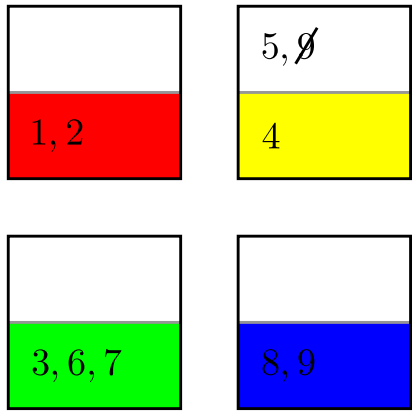


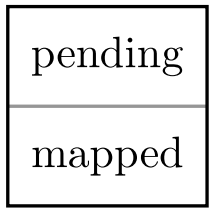




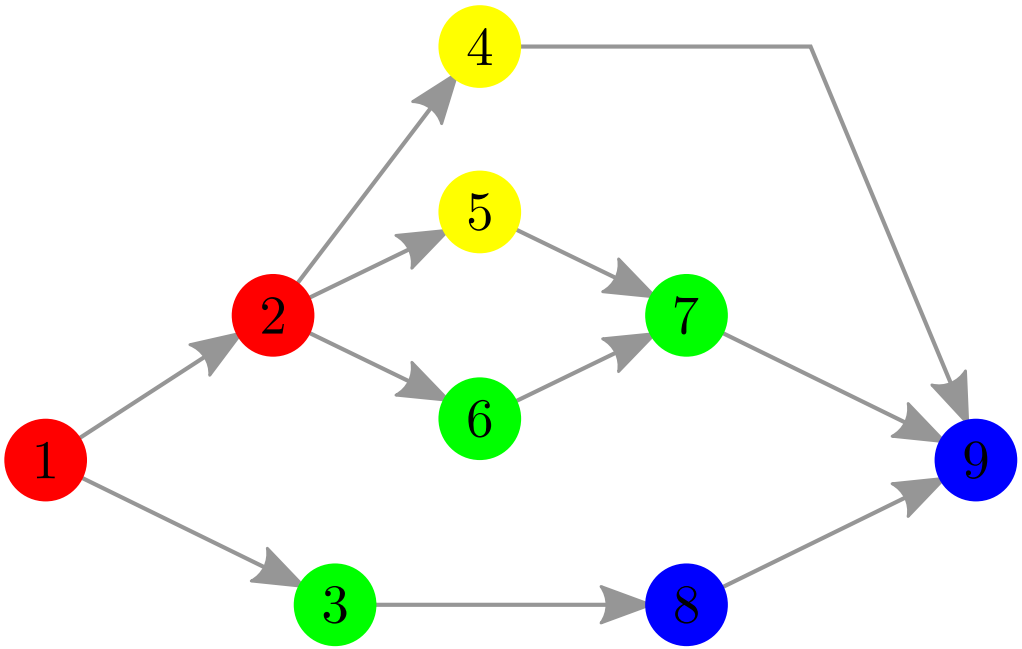
Greedy

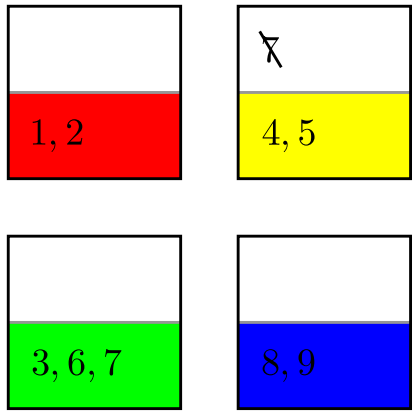


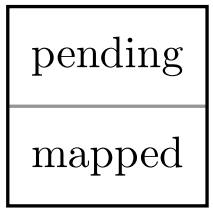




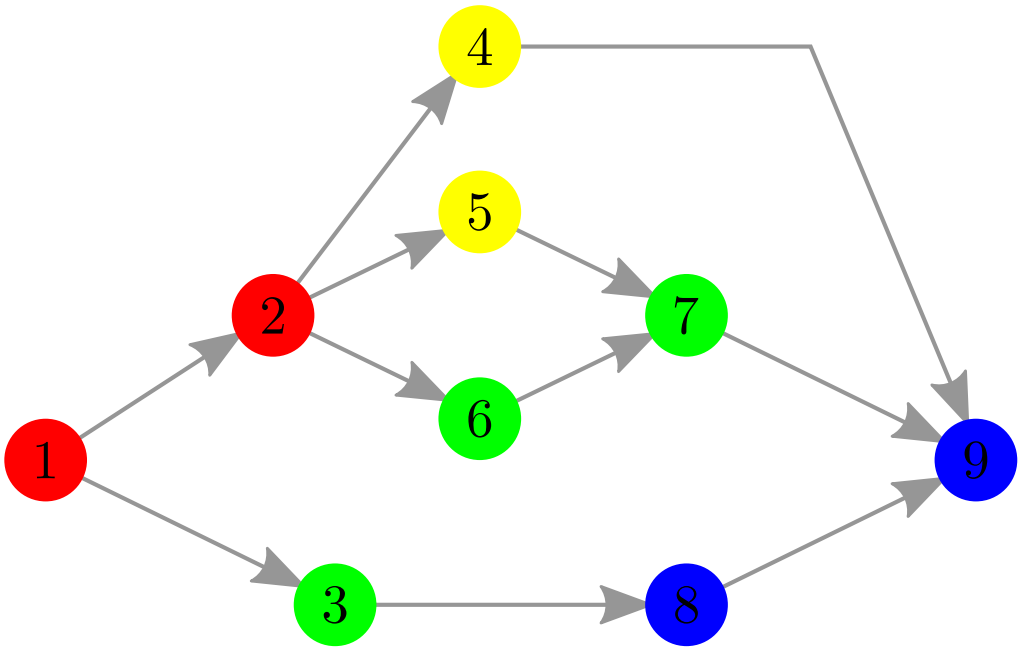
Greedy

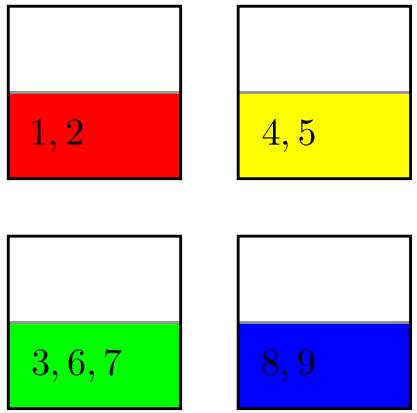


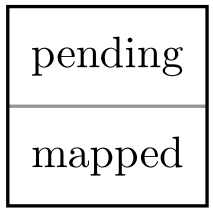




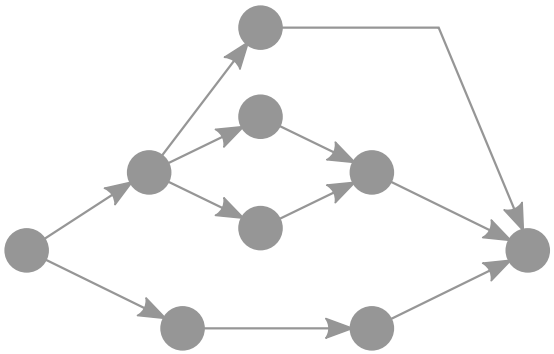
Greedy



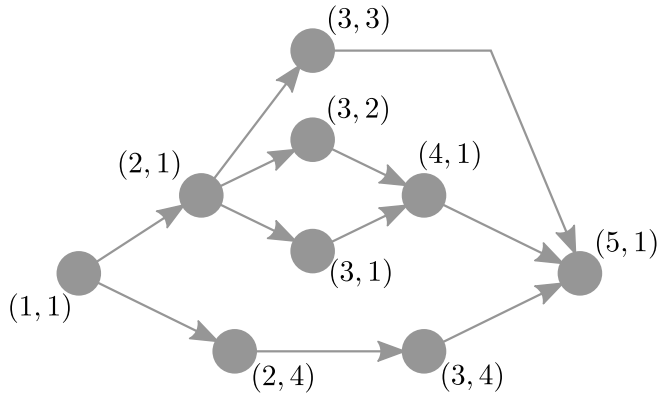




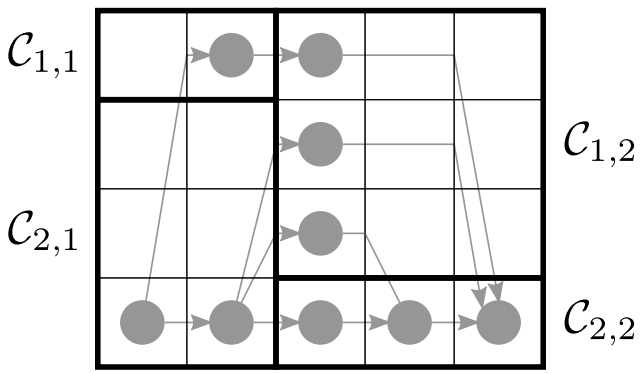
DPA2D



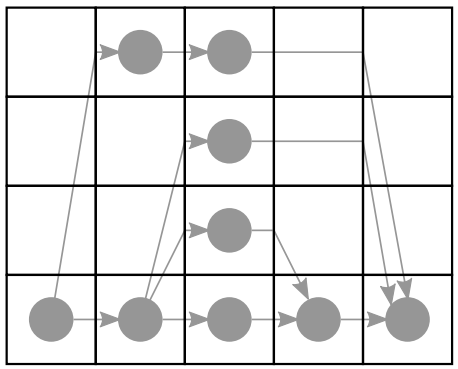
label



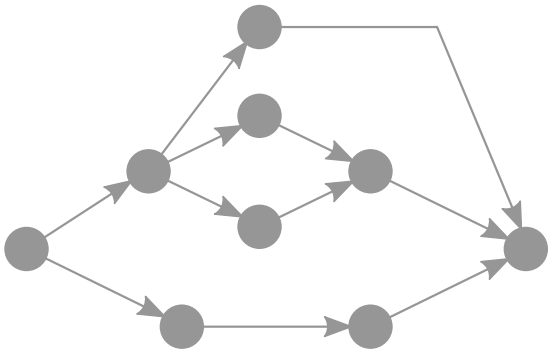
reorganize



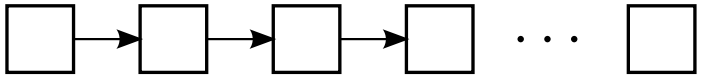
map



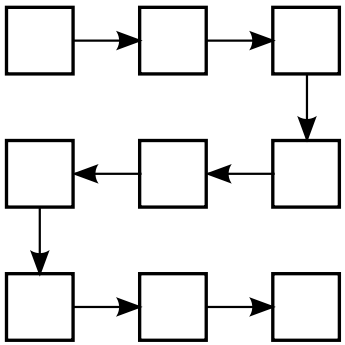
DPA1D, DPA2D1D



map



reconfigure

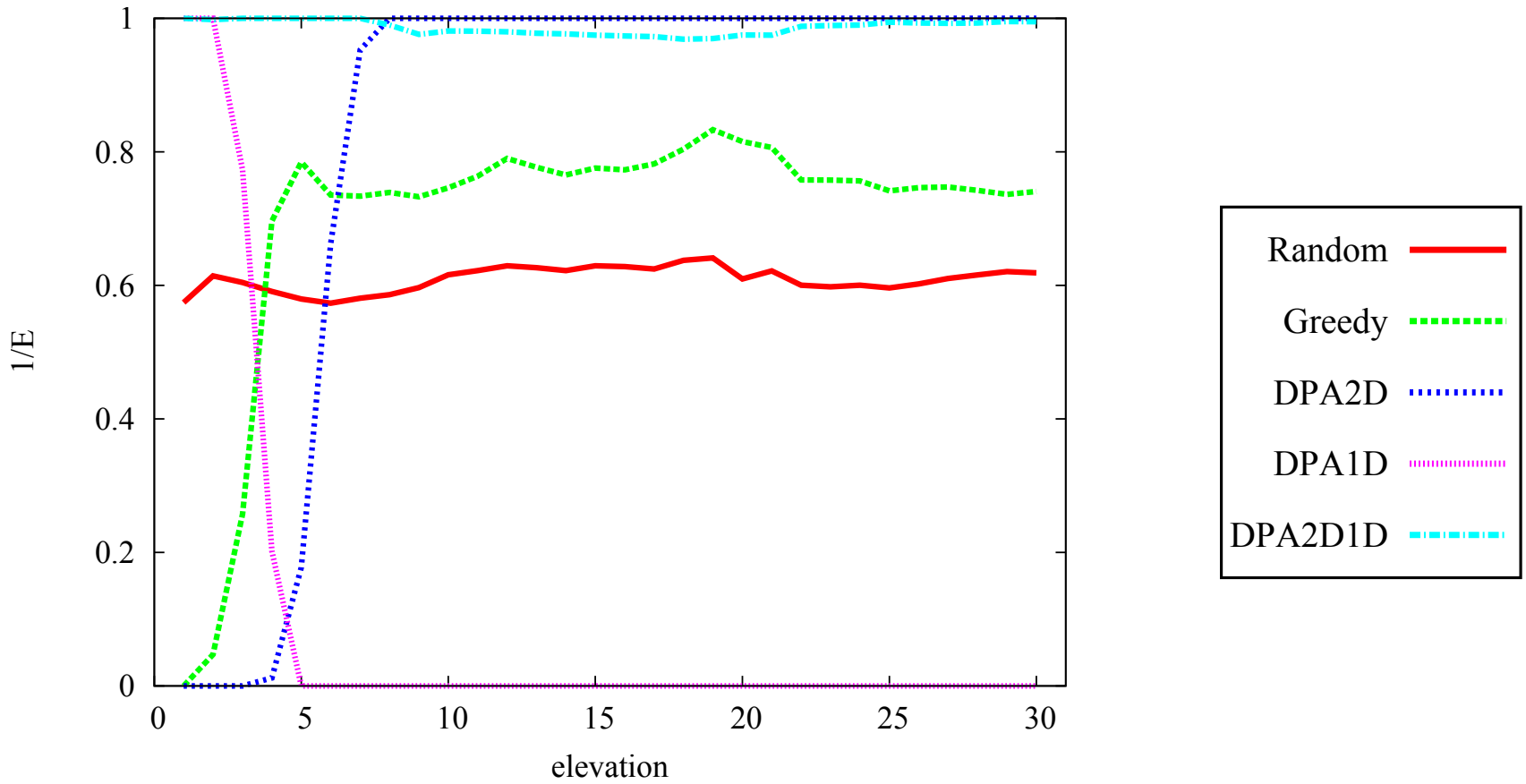


Outline of the talk

Simulation settings

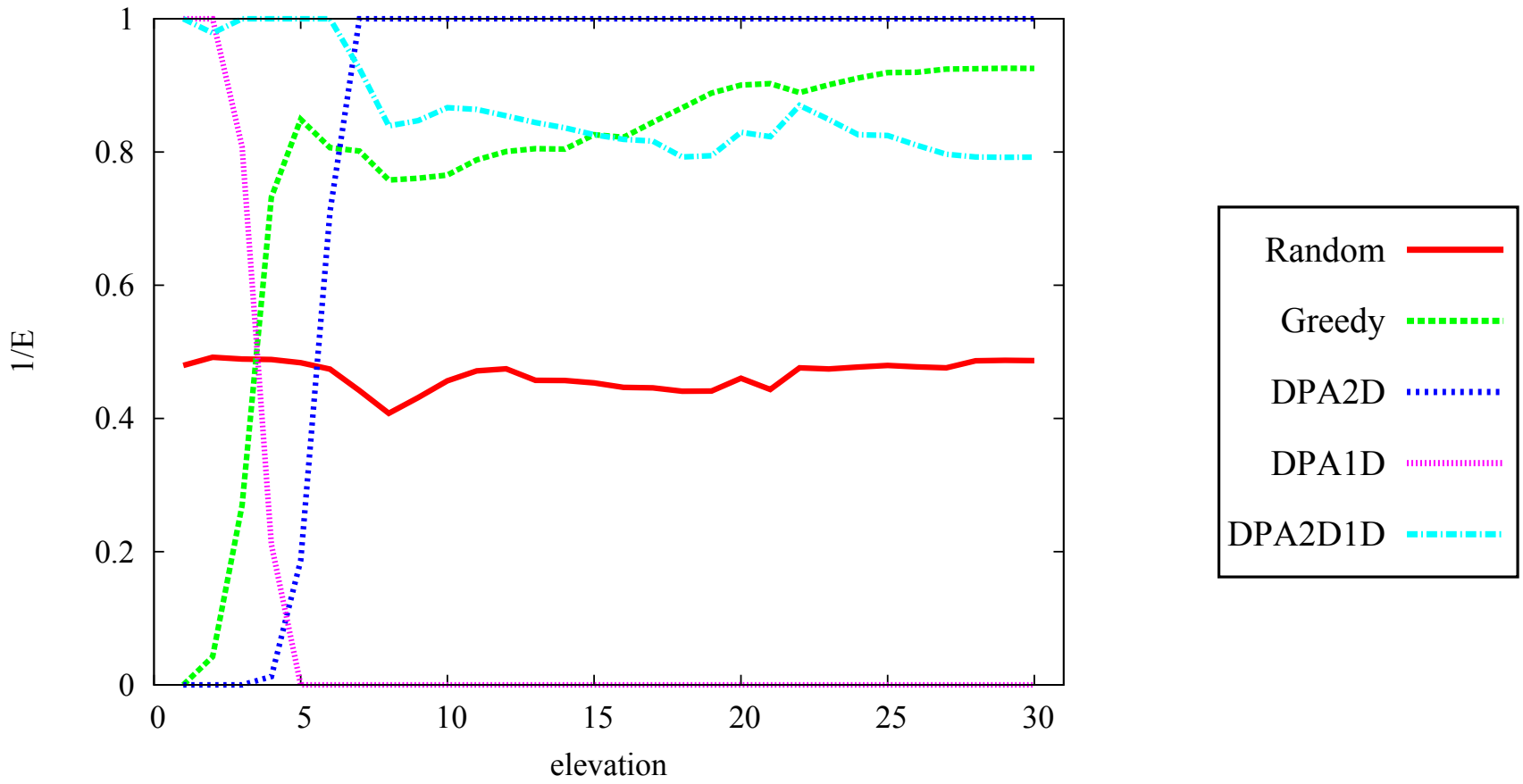
- Random SPGs
  - Averaged over 100 applications
  - 150 nodes
  - Elevation: from 1 to 30
- Real-life SPGs
  - 12 workflows
  - From 8 to 120 nodes
  - Elevation: from 1 to 17
- Varying communication-to-computation ratio

Random SPGs; computation intensive



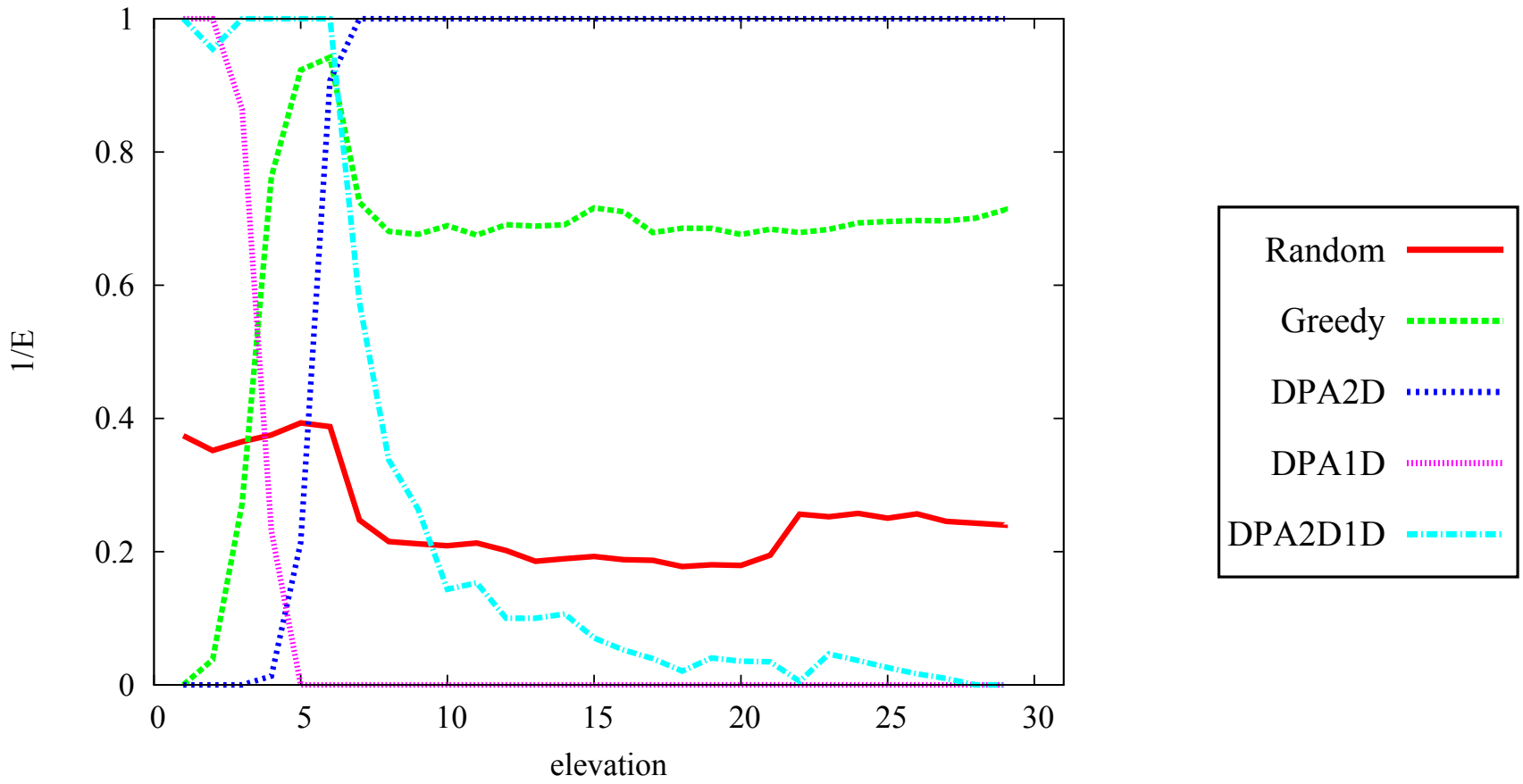
- **DPA1D** best then very rapidly useless
- **DPA2D** bad then rapidly best
- **DPA2D1D** always close to best
- **Greedy** intermediate

Random SPGs; balanced



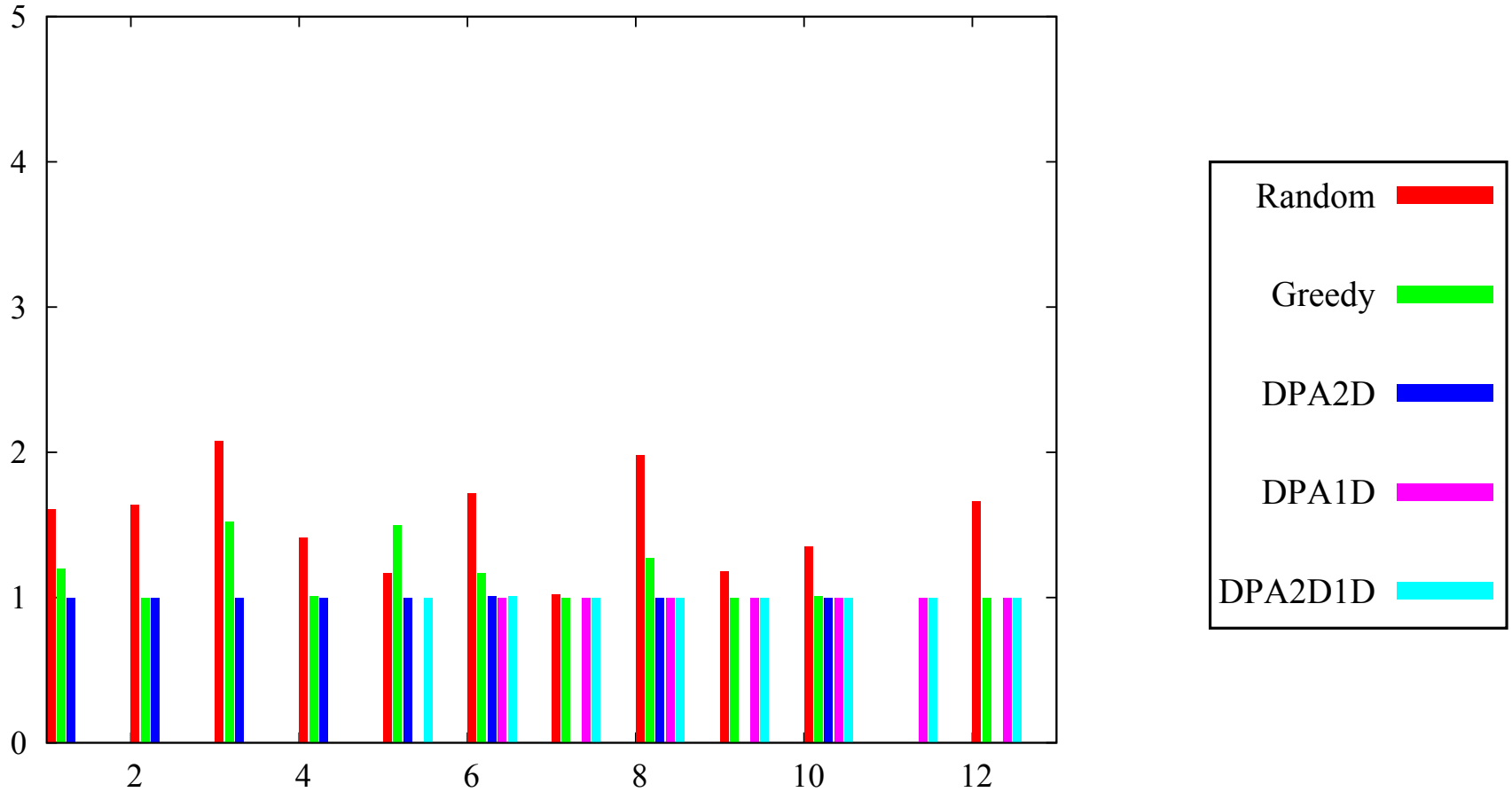
- **DPA1D**, **DPA2D** and **Greedy** better compared to **Random**
- **DPA2D1D** crumbles

Random SPGs; communication intensive

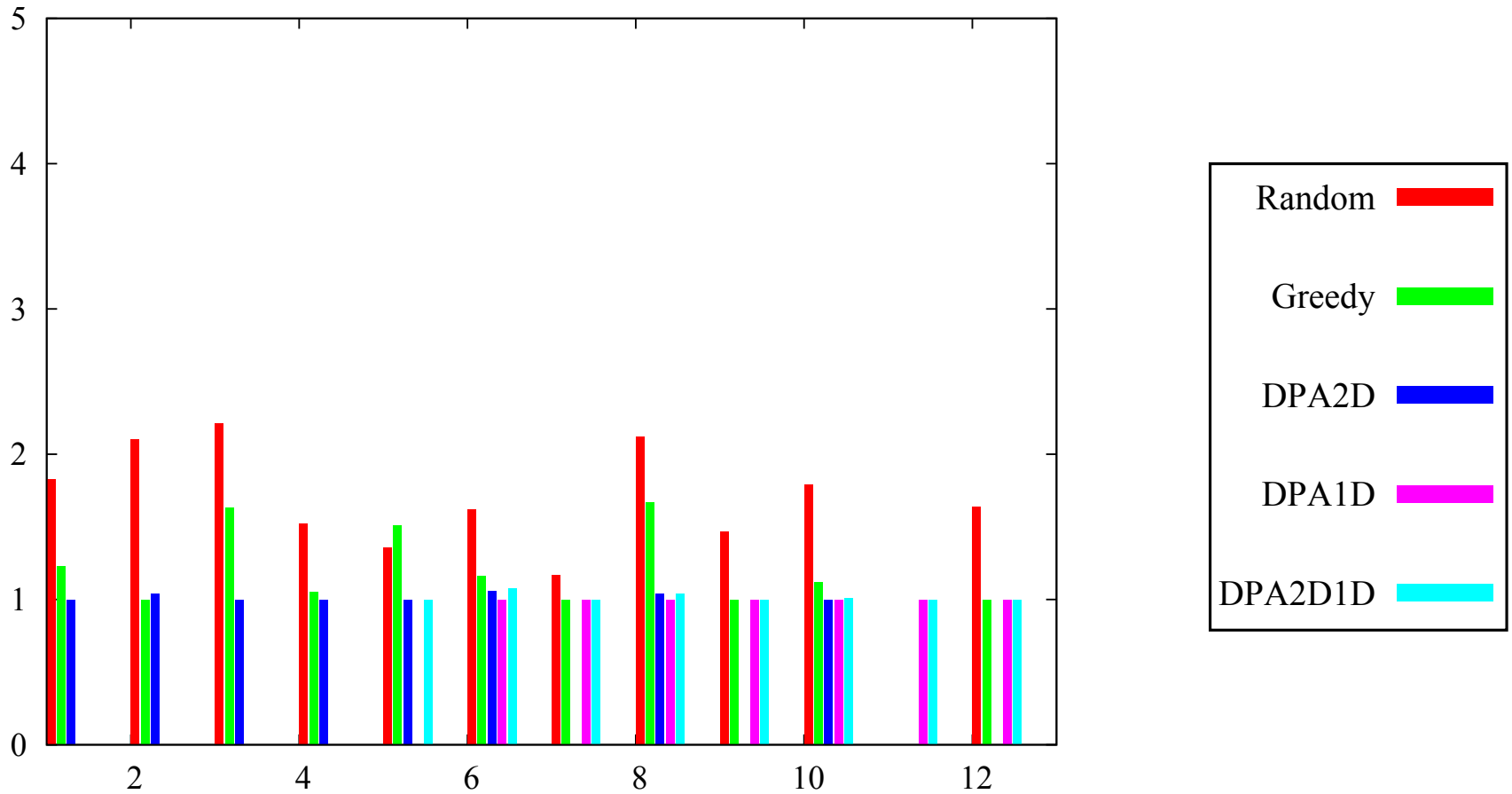


- **DPA1D**, **DPA2D** and **Greedy** better compared to **Random**
- **DPA2D1D** crumbles

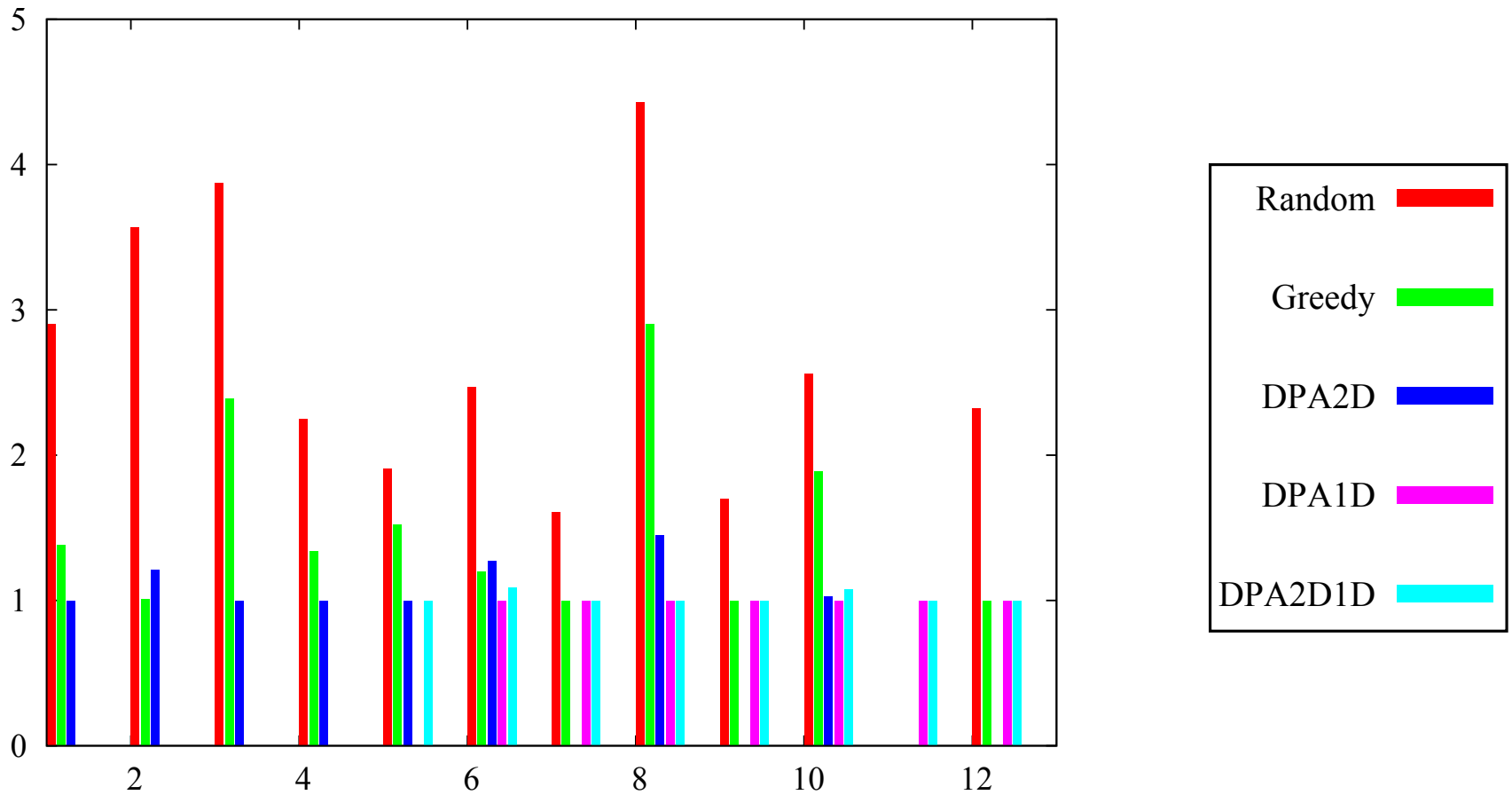
StreamIT; computation intensive



StreamIT; balanced



StreamIT; communication intensive



Conclusion

- Exhaustive complexity study
- Efficient heuristics for all kinds of SPGs
- Simulations on both randomly generated and real-life SPGs
- Future work: simplified ILP to assess the absolute performance of the heuristics, more accurate power consumption model for communications, multi-path routing

Questions

?