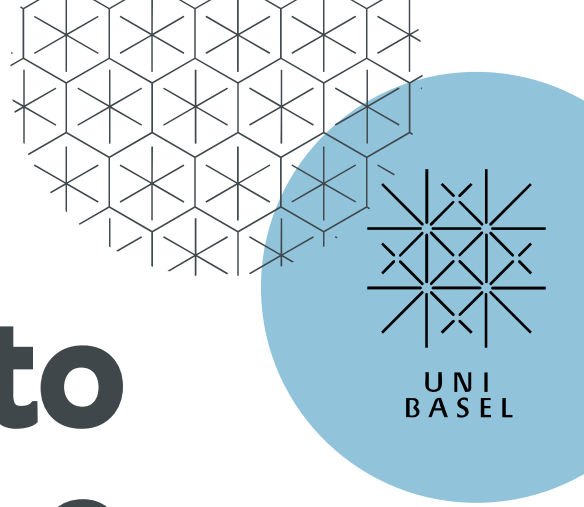


To Merge or to Cost Partition?

by Miruna Muntean

Examiner: Gabriele Röger
Supervisor: Thomas Keller

Table of contents

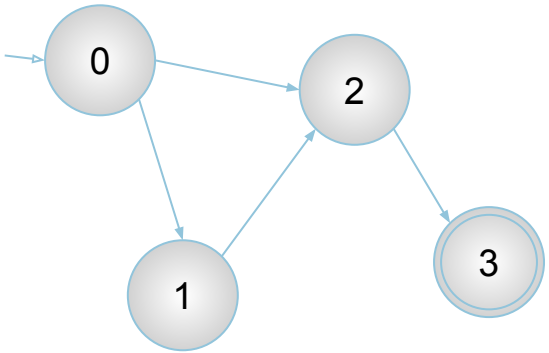
Menu

01

Introduction

Objective

- *Planning* = a sequence of actions aiming to accomplish the target
- Huge *state space*



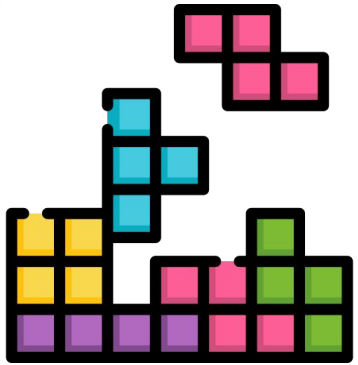
Assuming all path costs are 1:

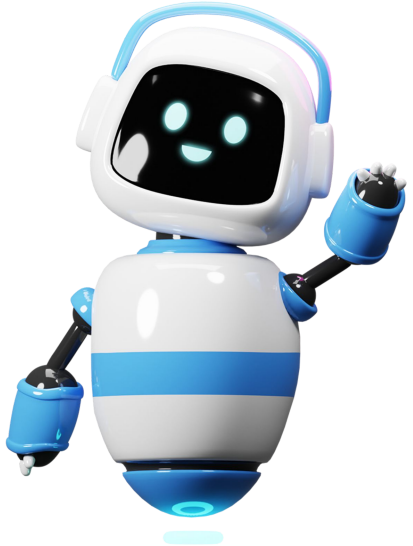
➔ 0 –> 1 –> 2 –> 3 (possible path)

➔ 0 –> 2 –> 3 (*cheapest* path)

- Interested in the **cheapest path** for optimal planning

Planning in Our Lives





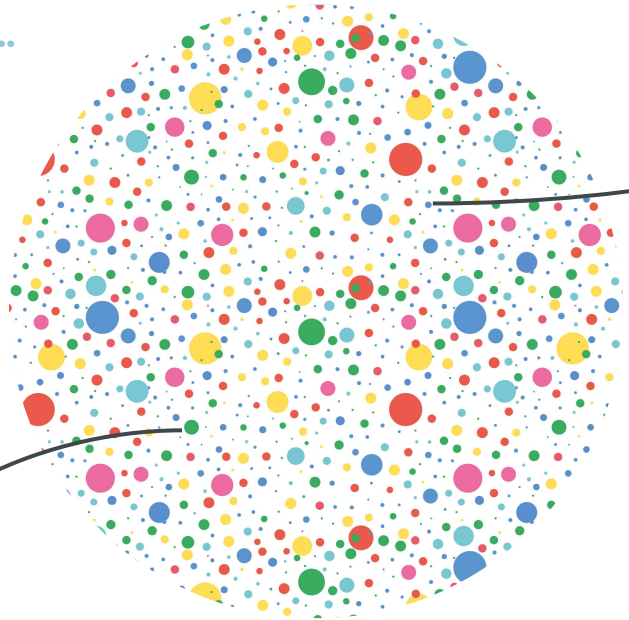
Menu

02

Key Concepts

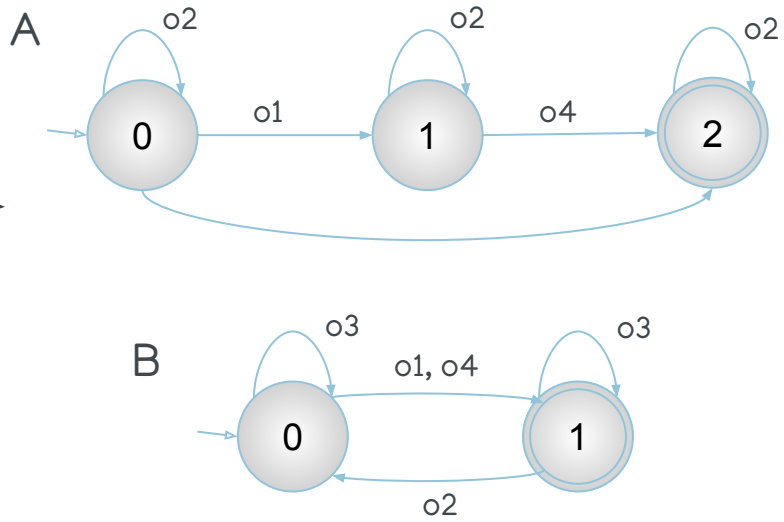
Vast Field of Algorithms

2 different algorithms…
among many others:



Merge-and-Shrink

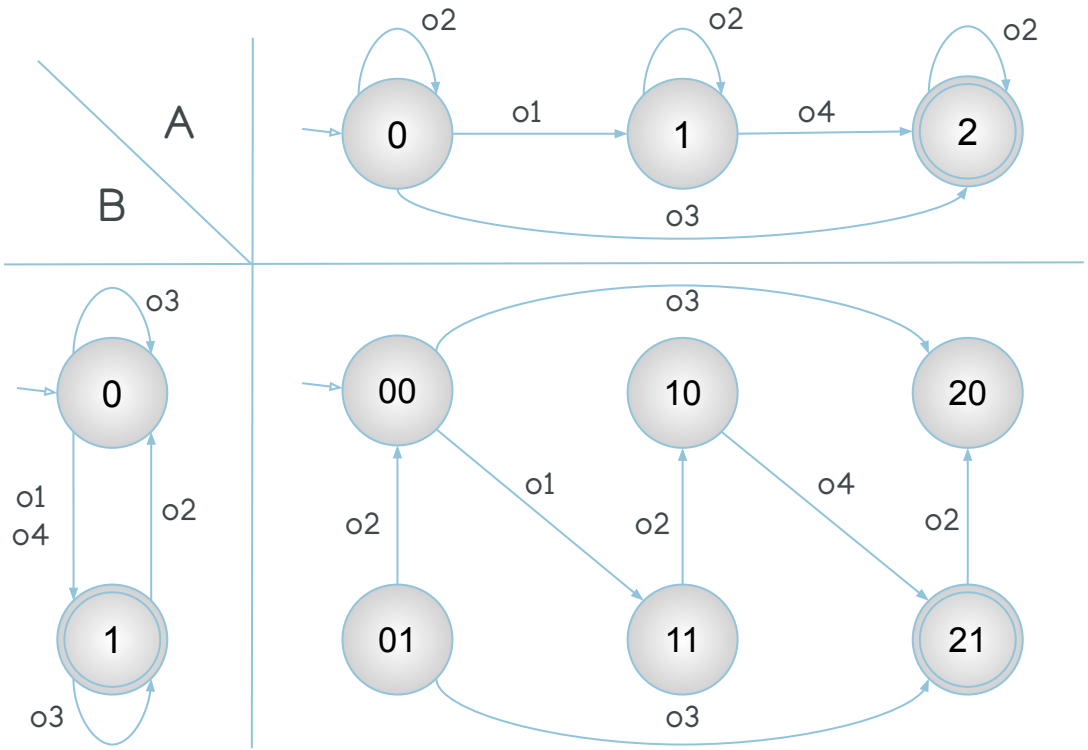
- Used to generate well-informed **abstraction heuristics**
- Starts from atomic projections



- **Steps:**
  1. Merge
  2. Shrink
  3. Prune
  4. Label Reduction

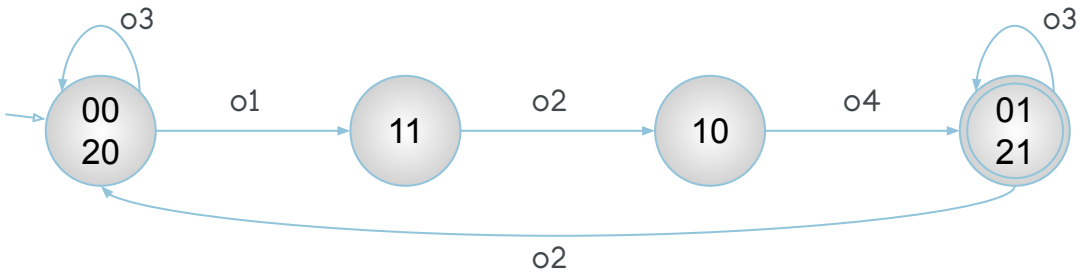
Merge

= Perform synchronised product of transition systems



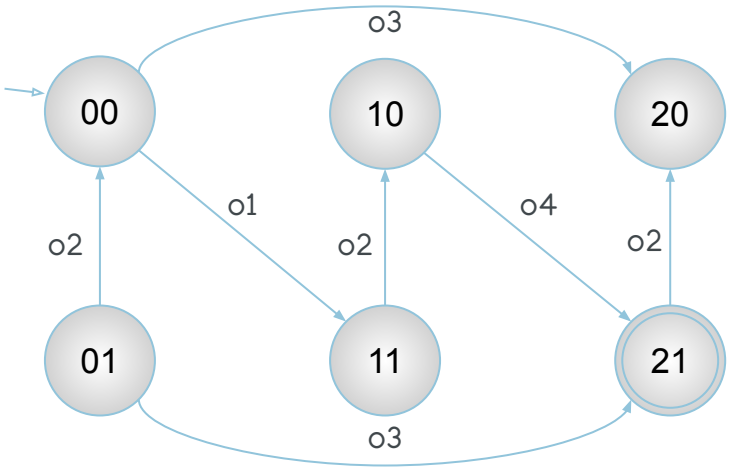
Shrink

= Reduce the size of a single factor by abstraction



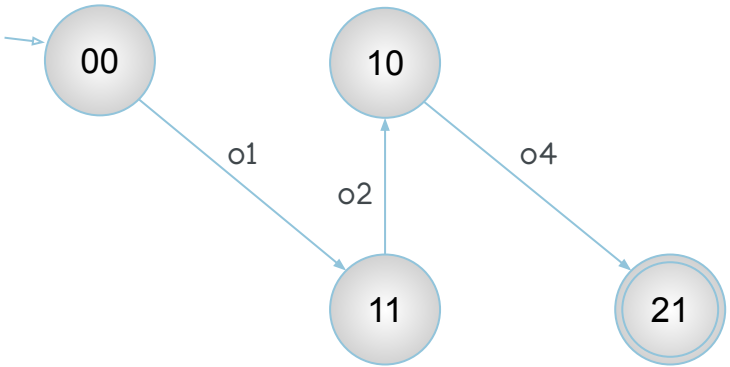
Prune

= Remove irrelevant and unreachable states

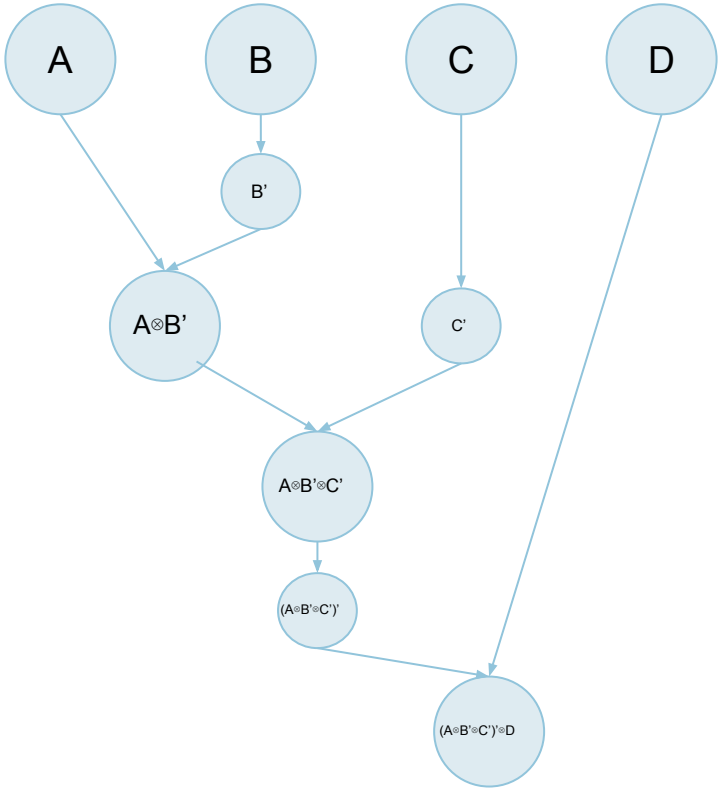


Prune

= Remove irrelevant and unreachable states



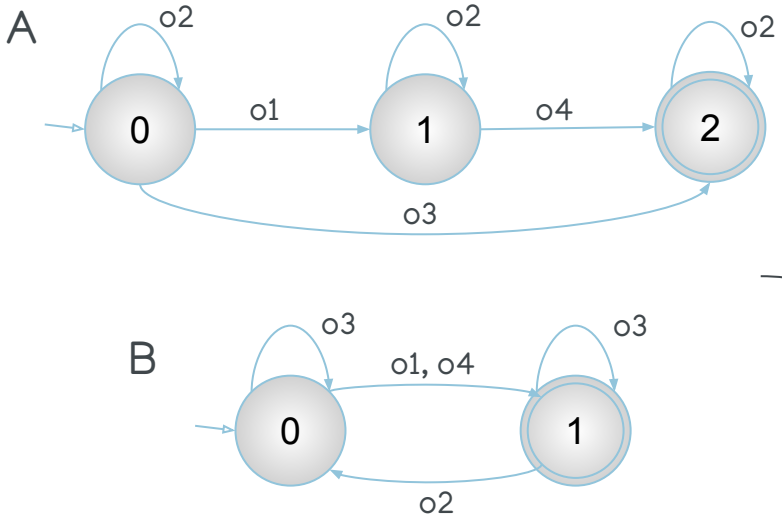
Merge-and-Shrink



Cost Partitioning

= Admissibly combining multiple heuristic values

- Optimal cost partitioning is *very expensive* to compute in practice



- Best known in practice:
  saturated cost partitioning
  - Quick sub-optimal cost partitioning
  - Results depend on heuristic order

$cost_A$:	$cost_B$:
o1 → 0.5	o1 → 0.5
o2 → 1	o2 → 0
o3 → 1	o3 → 0
o4 → 0.5	o4 → 0.5

Menu

03

Implementation

Combining the Key Concepts

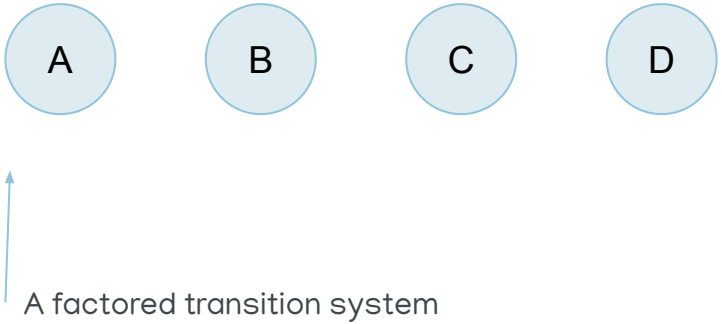
Main idea:

- Start by computing the atomic projections
- Calculate the quality of merging compared to cost partitioning
- If merging is more informative, merge
- Else, cost partition
- When no merging is more informative, stop

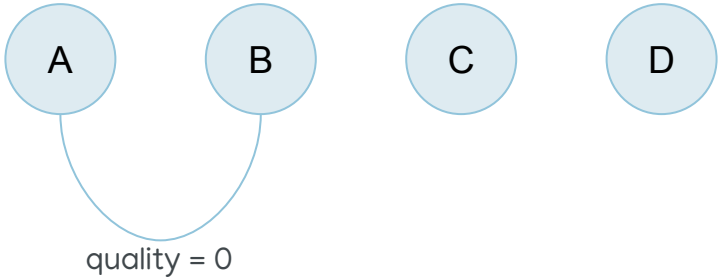
Quality-based algorithm

➔ *Merge-and-shrink*: always one system in the end
➔ *Cost partitioning*: always all systems in the end
➔ *This algorithm*: in-between the 2 above

The Algorithm

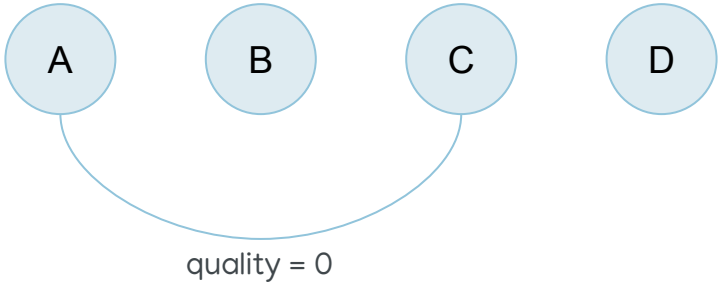


The Algorithm



A⊗B –> 0

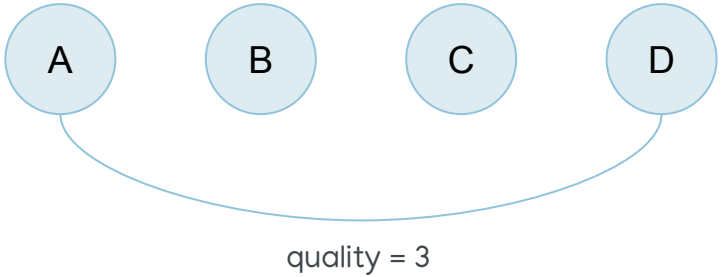
The Algorithm



A⊗B –> 0

A⊗C –> 0

The Algorithm

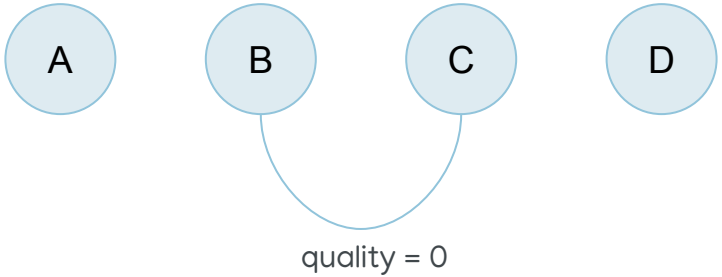


A⊗D –> 3

A⊗B –> 0

A⊗C –> 0

The Algorithm



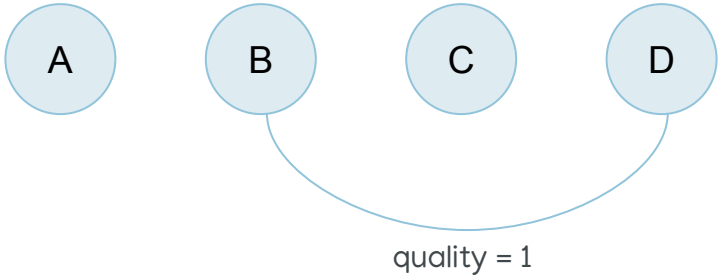
A⊗D –> 3

A⊗B –> 0

A⊗C –> 0

B⊗C –> 0

The Algorithm



A⊗D –> 3

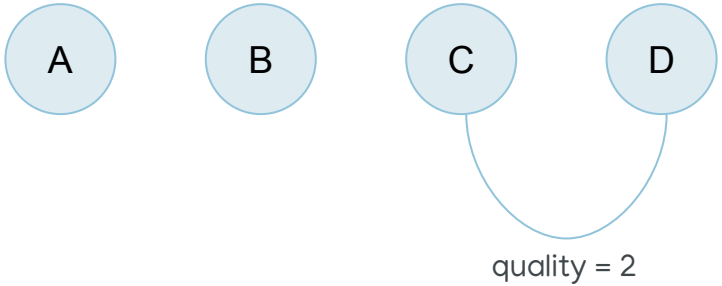
B⊗D –> 1

A⊗B –> 0

A⊗C –> 0

B⊗C –> 0

The Algorithm



A⊗D –> 3

C⊗D –> 2

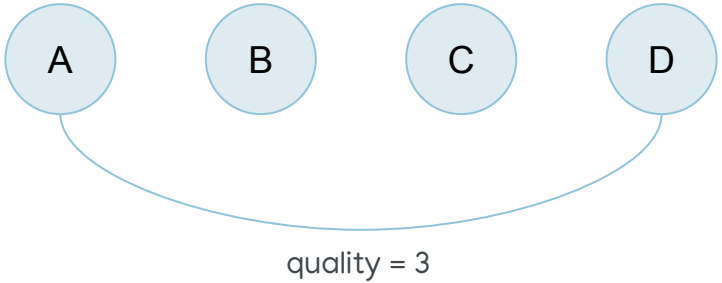
B⊗D –> 1

A⊗B –> 0

A⊗C –> 0

B⊗C –> 0

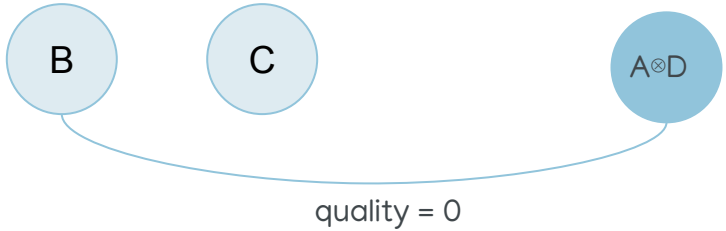
The Algorithm



- A and D are now merged as:
  - Merging brings us more value than cost partitioning
  - The size is within the imposed limit

- The process starts again
  - A and D are removed and replaced by their merged product

The Algorithm



A⊗D –> 3

C⊗D –> 2

B⊗D –> 1

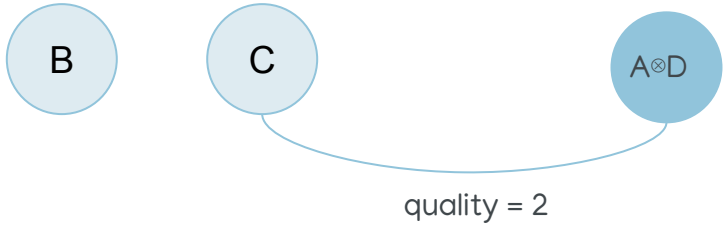
A⊗B –> 0

A⊗C –> 0

B⊗C –> 0

B⊗(A⊗D) –> 0

The Algorithm

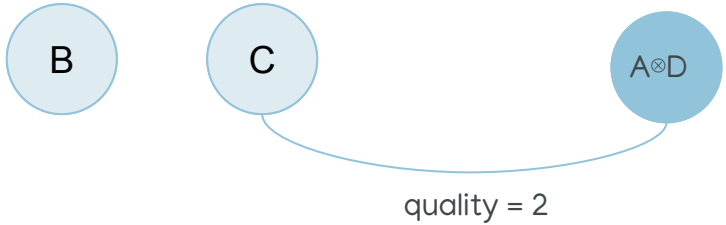


C⊗(A⊗D) –> 2

B⊗C –> 0

B⊗(A⊗D) –> 0

The Algorithm



- C and A⊗D are now merged as C⊗(A⊗D) has the highest quality of 2

- The process starts again
  - C and A⊗D are removed and replaced by their merged product

The Algorithm

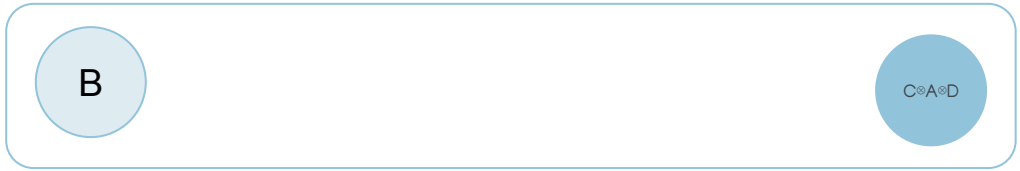
C⊗(A⊗D) –> 2

B⊗C –> 0

B⊗(A⊗D) –> 0

B⊗(C⊗(A⊗D)) –> 0

The Algorithm



cost partition

C⊗(A⊗D) –> 2

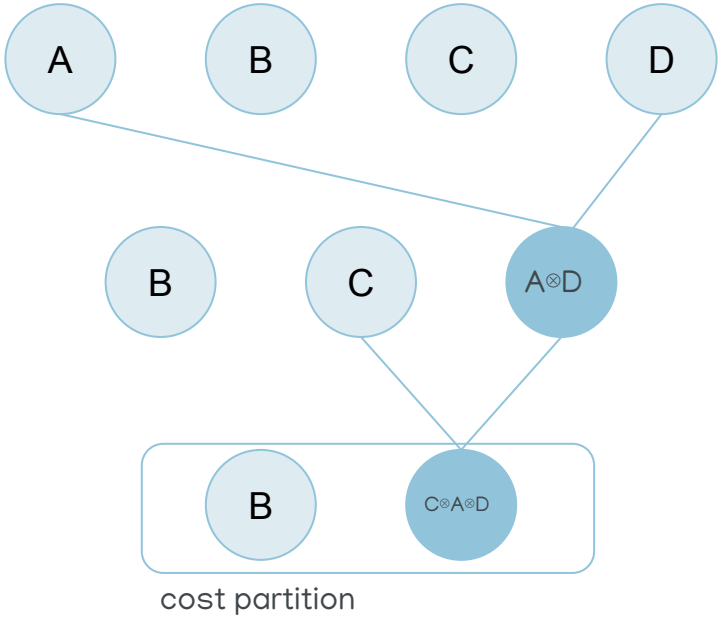
B⊗C –> 0

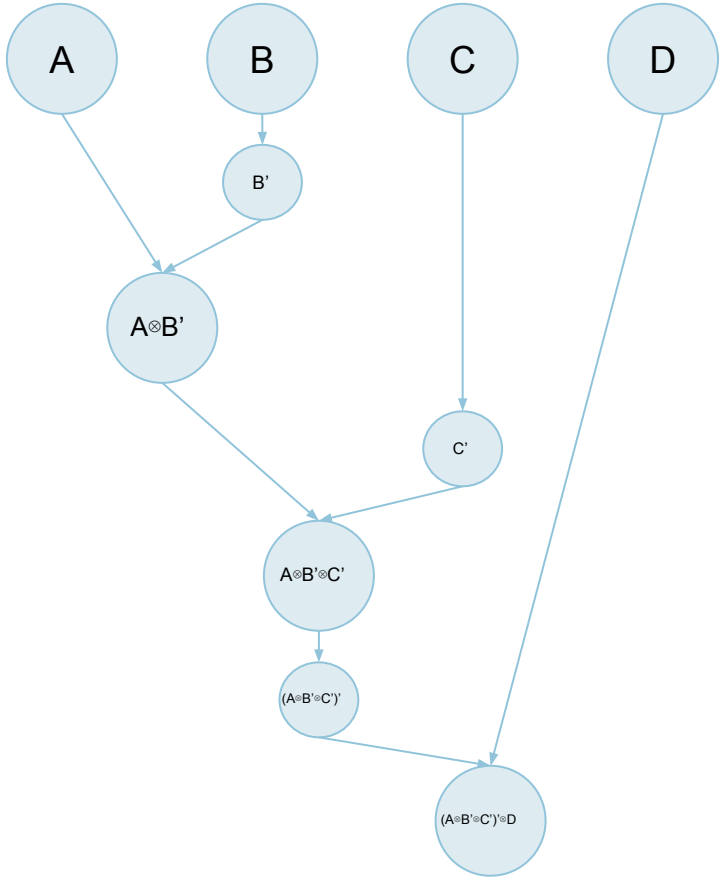
B⊗(A⊗D) –> 0

B⊗(C⊗(A⊗D)) –> 0

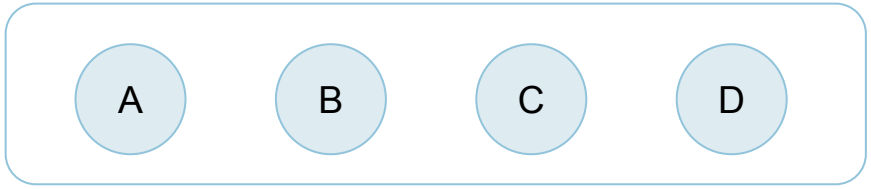
- All qualities are 0 => **cost partitioning** over the remaining atomic projections
- 2 projections are left, not only 1

The Algorithm



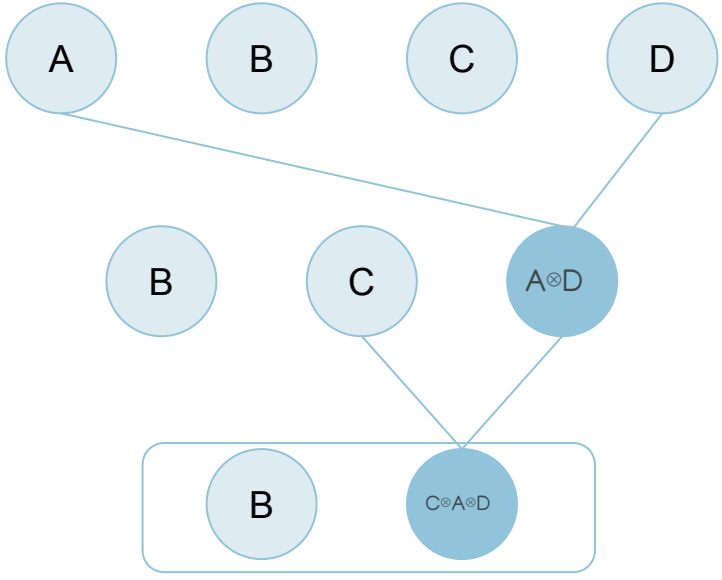


Merge-and-shrink



cost partition

Cost partitioning



cost partition

MSCP

Quality

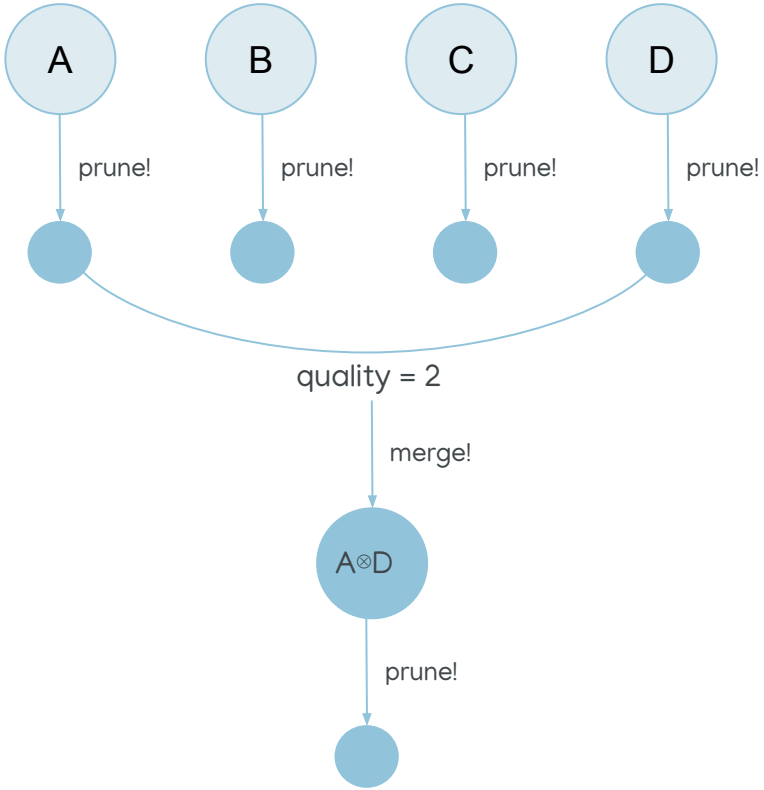
$$quality = h^{\alpha} \otimes h^{\beta} - h^{\alpha+\beta}$$

- For two abstractions α and β with heuristics h^{α} and h^{β}, where
  - $h^{\alpha} \otimes h^{\beta}$ is the heuristic of the synchronised product
  - $h^{\alpha+\beta}$ is the heuristic of the cost partitioning

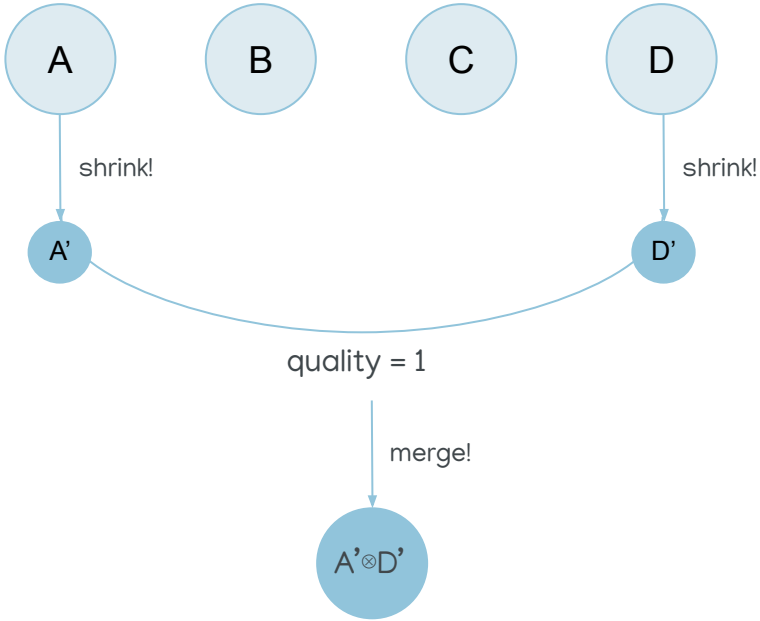
- Shows the degree of increase in information brought by using merge-and-shrink

  "How much more does merging brings in?"

Pruning



Shrinking



Menu

04

Experiments

Analysis Aspects

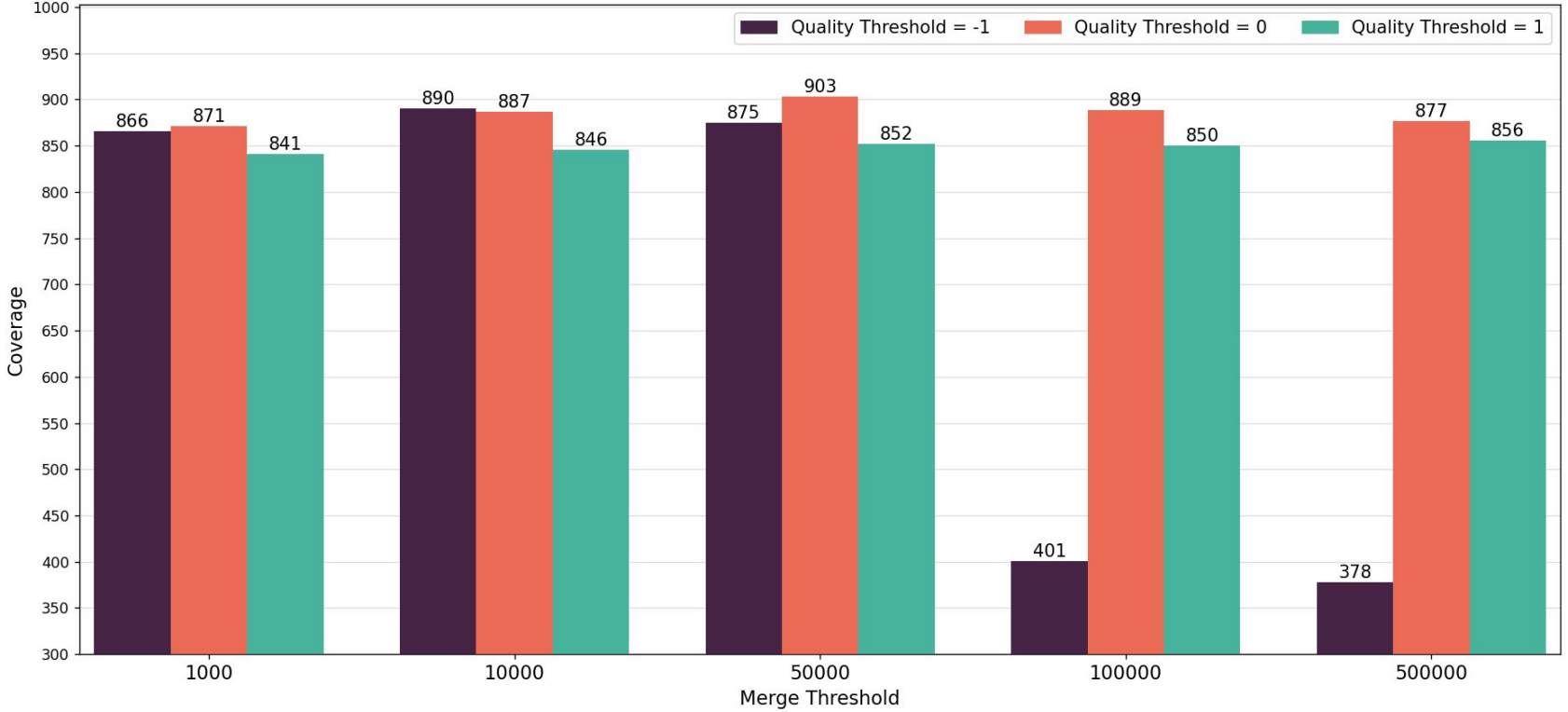
Check 3 main aspects:

- Coverage
- Expansions
- Number of merges

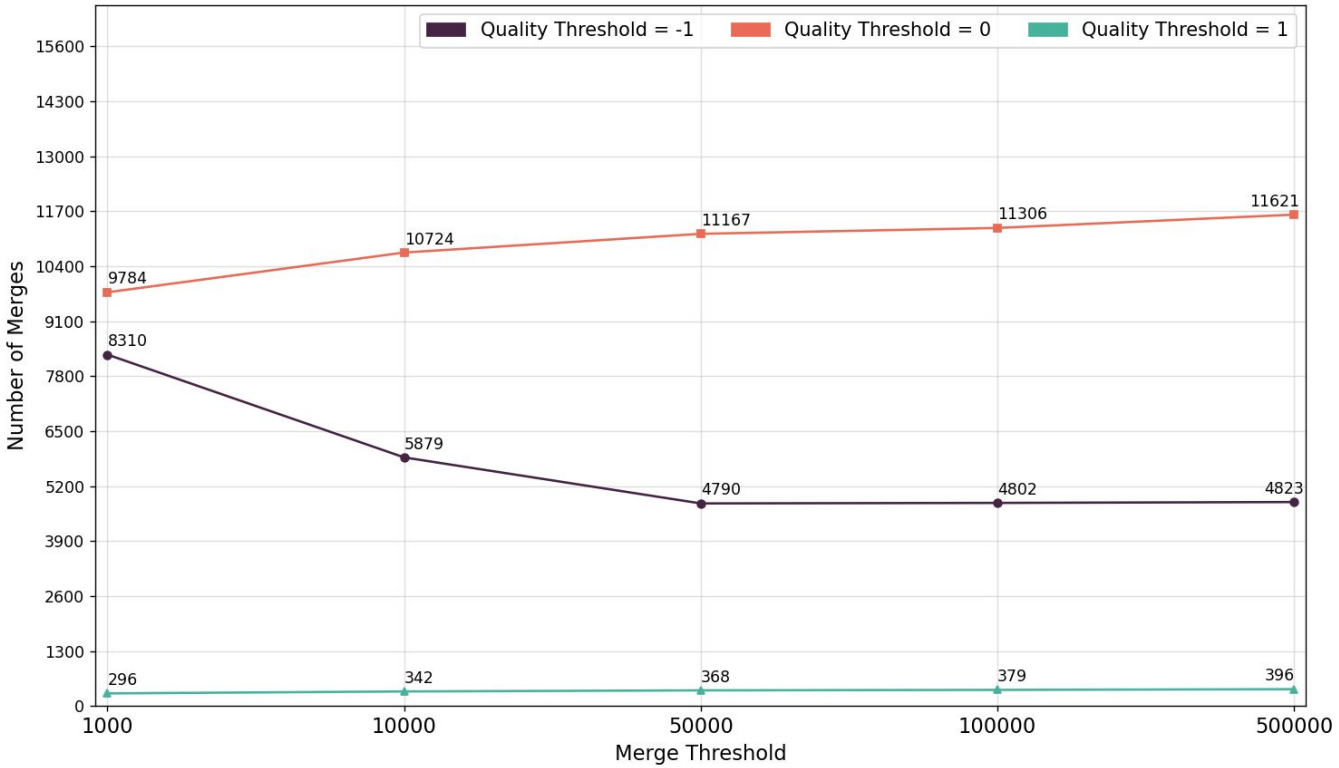
Based on:

- **Merge** threshold
  - How large do we allow a merge to be?
- **Quality** threshold
  - How much more informative does merging need to be?

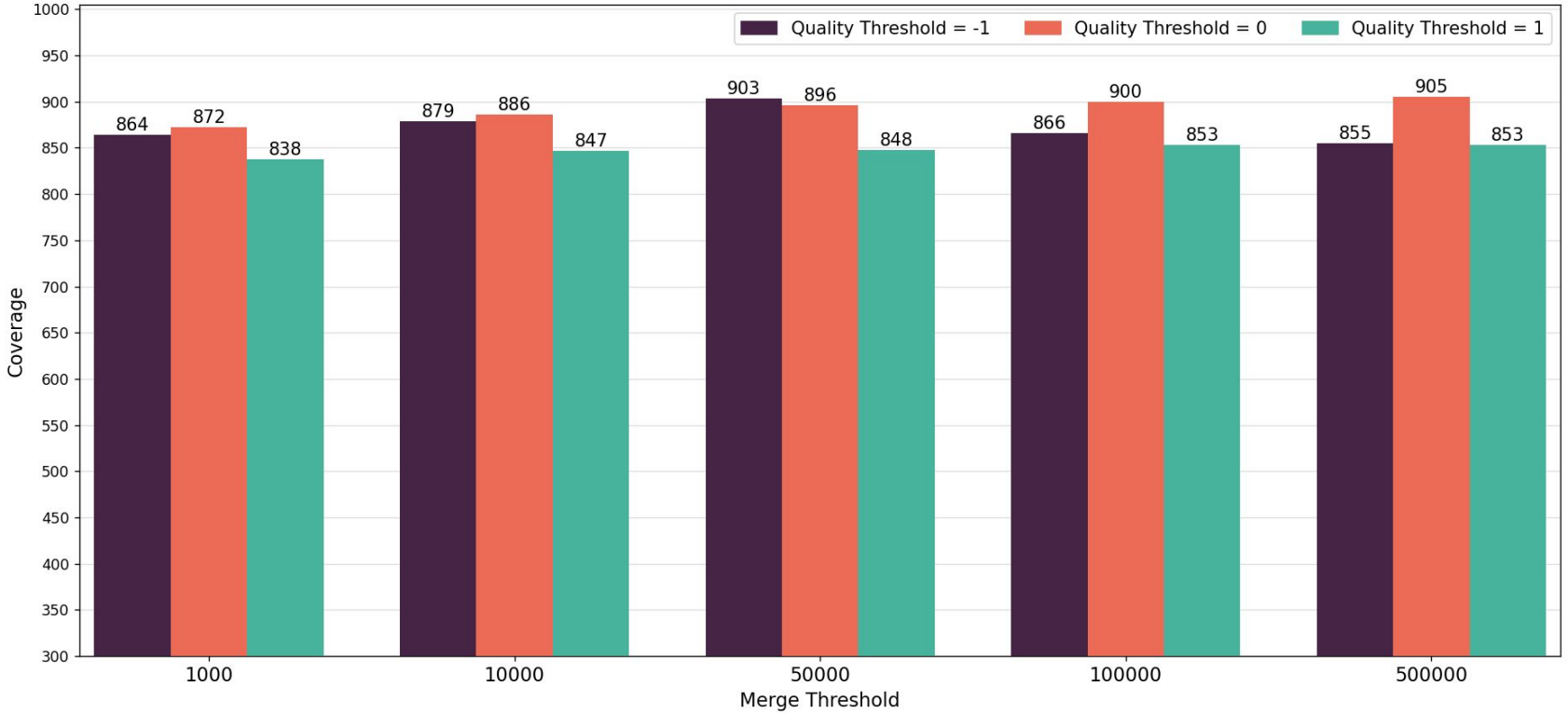
The Baseline (Merge vs Cost Partition)



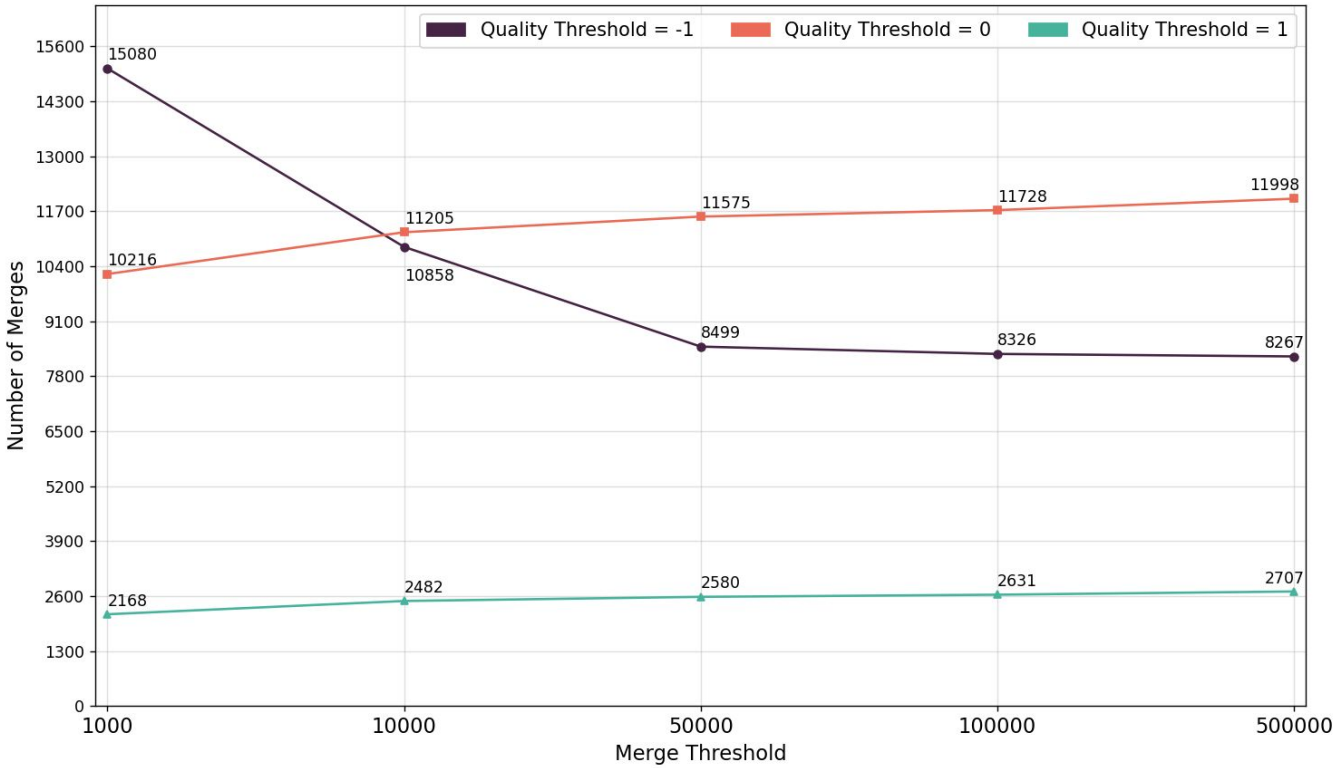
The Baseline (Merge vs Cost Partition)



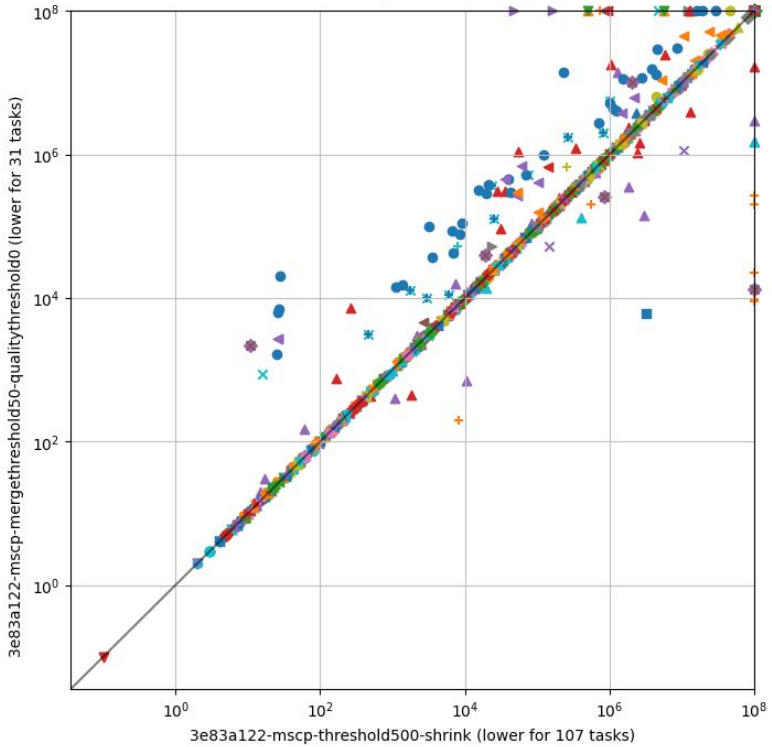
Adding Pruning and Shrinking



Adding Pruning and Shrinking



Adding Pruning and Shrinking

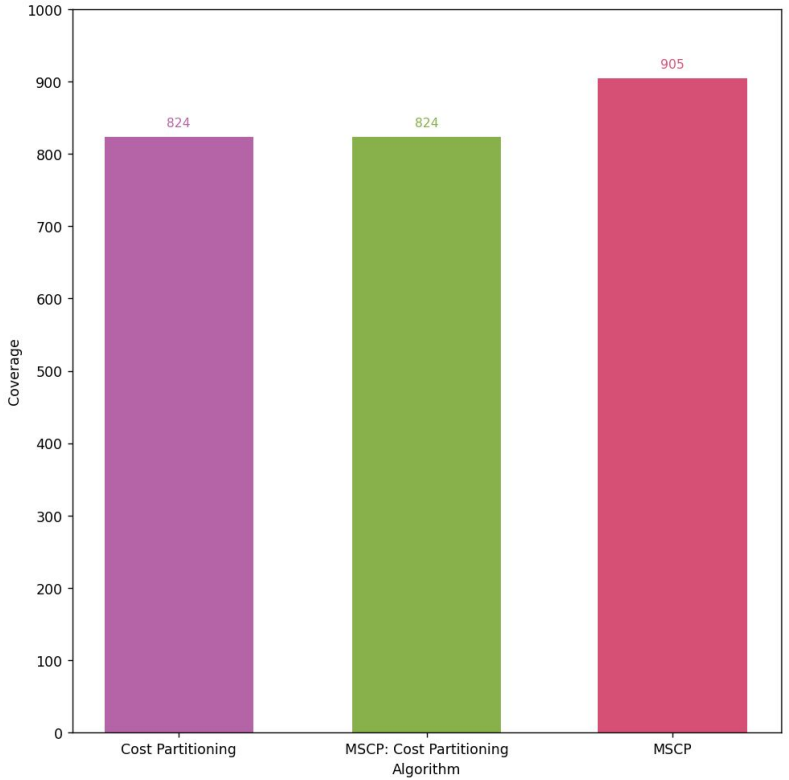


Comparing to Cost Partitioning

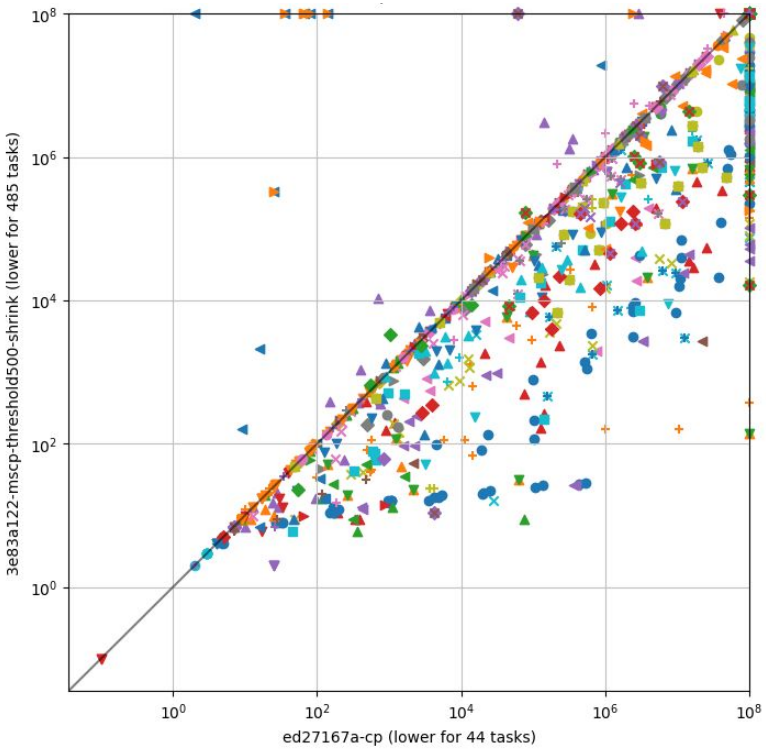
- The algorithms can reproduce cost partitioning by setting:
  - merge threshold as –1
  - quality threshold as infinity
  - other parameters to *false*

- As theoretically expected:

  Number of merges always zero for cost partitioning



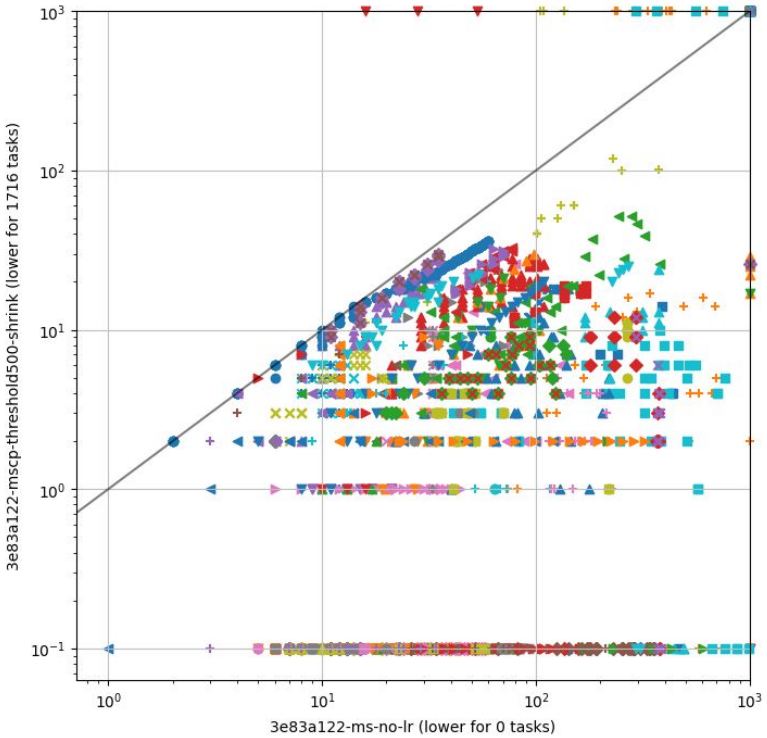
Comparing to Cost Partitioning



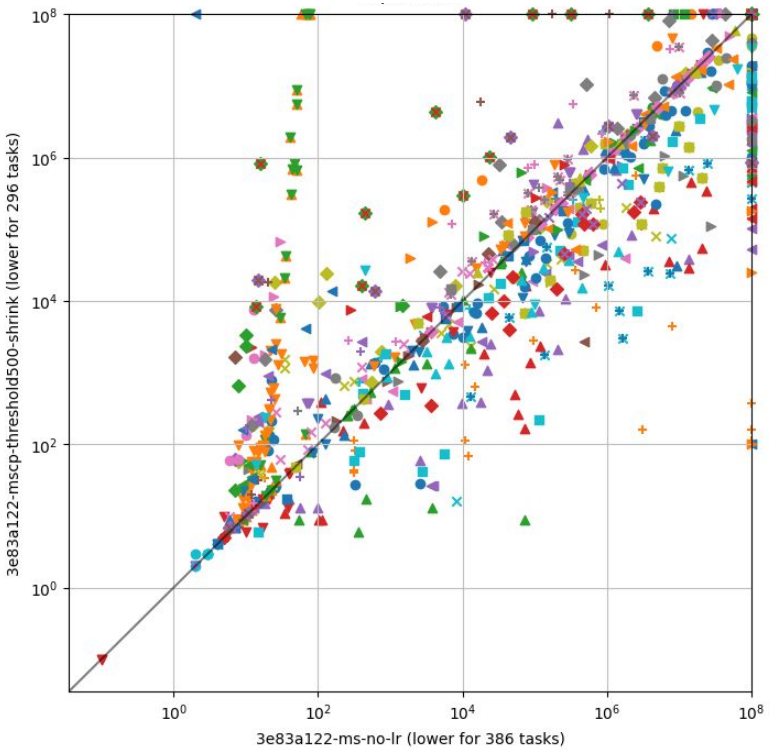
Comparing to Merge-and-Shrink

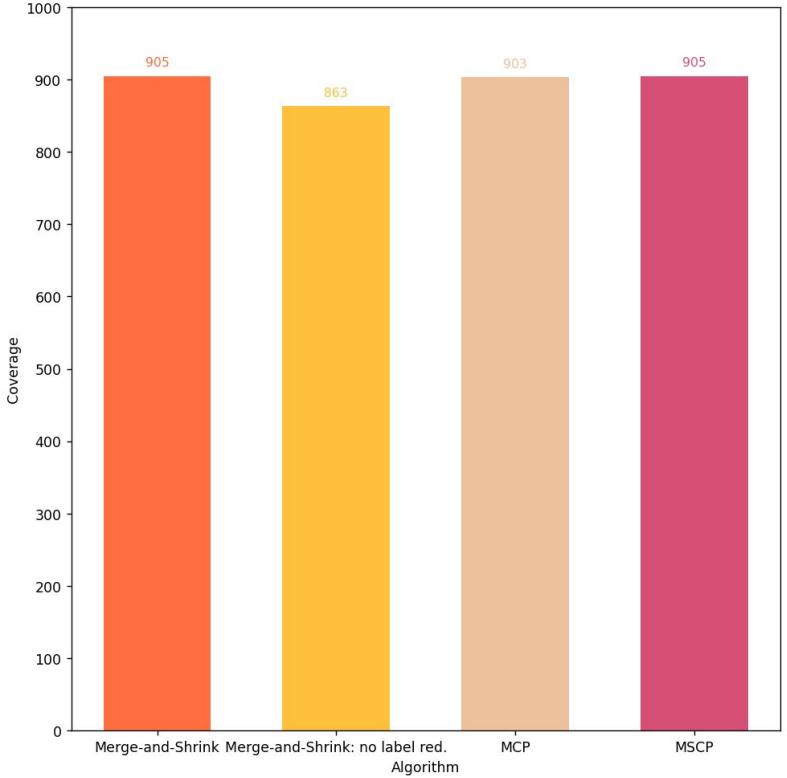
- Can only partially reproduce merge-and-shrink algorithm:
  - label reduction not implemented
  - no merge-strategy involved
- Merge-and-shrink used with and without label reduction
- As theoretically expected:

  Number of merges always higher for merge-and-shrink



Comparing to Merge-and-Shrink





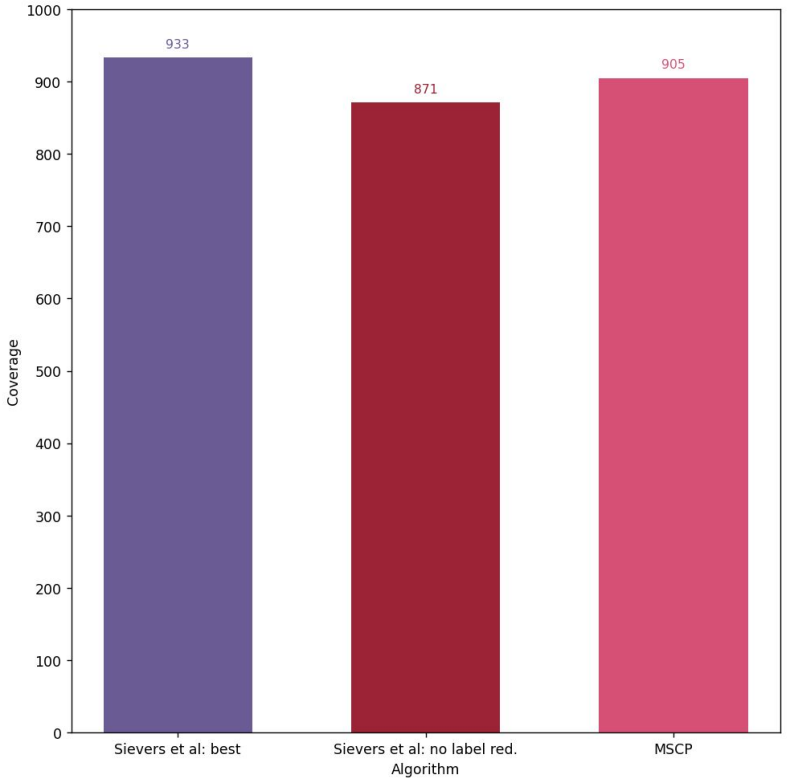
Comparing to Merge-and-Shrink

	M&S	M&S – no label reduction	MCP (baseline)	MSCP
coverage	905	863	903	905
number of merges	82,000	136,425	12,848	13,745
total time	4.03	2.21	0.97	1.59

Comparing to Previous Work

- *Label reduction* = very important
  - Different approach
  - Same goal

	Sievers et al: best	Sievers et al: no lab red	MSCP
coverage	933	871	905
total time	4.33	3.88	1.80



Quality Enhancing (Max Occurrences)

- Sometimes we can have all qualities as 0, such as:
  {B⊗(A⊗D) –> 0, B⊗G –> 0, G⊗H –> 0, C⊗H –> 0, B⊗C –> 0, ...}
- However, the next step could bring us a better quality
- But, the algorithm would stop merging and opt for cost partitioning

- **Solution:** Allowing occurrences of minimal quality
  - Fixed number of allowed cases of minimum quality
  - *Example*: if we set this to 2, there will be 2 iterations allowing quality = 0

=> **More merges** take place
=> **Larger initial heuristic** value

Quality Enhancing (Consider Size)

$$quality = \begin{cases} \infty, & \text{if } \mid \alpha \times \beta \mid \leq \mid \alpha \mid + \mid \beta \mid \text{ and } h^{\alpha} \otimes h^{\beta} \geq h^{\alpha+\beta} \\ 1, & \text{if } \mid \alpha \times \beta \mid \leq \mid \alpha \mid + \mid \beta \mid \text{ and } h^{\alpha} \otimes h^{\beta} = h^{\alpha+\beta} \\ h^{\alpha} \otimes h^{\beta} - h^{\alpha+\beta}, & \text{if } \mid \alpha \times \beta \mid \leq \mid \alpha \mid + \mid \beta \mid \text{ and } h^{\alpha} \otimes h^{\beta} < h^{\alpha+\beta} \\ \frac{h^{\alpha} \otimes h^{\beta} - h^{\alpha+\beta}}{\mid \alpha \times \beta \mid - (\mid \alpha \mid + \mid \beta \mid)}, & \text{otherwise} \end{cases}$$

- For two abstractions α and β with heuristics h^{α} and h^{β}, where
  - $h^{\alpha} \otimes h^{\beta}$ is the heuristic of the synchronised product
  - $|\alpha \times \beta|$ is the size of the synchronised product
  - $h^{\alpha+\beta}$ is the heuristic of the cost partitioning
  - $|\alpha| + |\beta|$ is the sum of the size of atomic projections

Menu

05

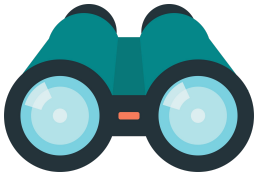
Future Work

Look-ahead Factor

- How much value a decision to merge brings in future iterations
- **Currently:** If all qualities are 0 => stop and cost partition

  What if next iteration would bring a valuable result?

- Maximum occurrences of minimal quality blindly solves this, but it does not actually consider if it is worth it...
  ... or how many times can we ignore a minimal value

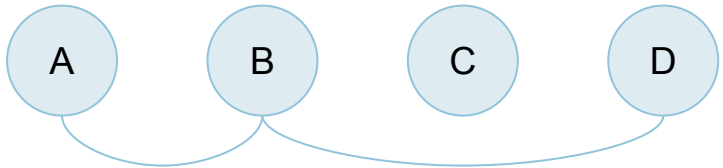


- **Solution:**
  - The possibility to calculate the quality of the next iteration
  - Based on the merging that took place in the current iteration

Non-Linear Merge Strategy

- Currently, once a transition system is merged, it becomes inactive
- All already existing data about it is not considered anymore
- And linearly looping through systems can be time-consuming

- *Pearks*:
  - Simulate parallelism
  - Merge a transition system more times
  - Useful in very large problems



- *Drawback*:
  - Potential massive memory usage

Menu

06

Conclusion

Conclusions

- The 2 main concepts can be combined with a simple, greedy, quality-based approach
- Label reduction is a very important step of merge-and-shrink for good results

- Similar coverage to merge-and-shrink, with and without improvements … but without label reduction!
- Less than half time and much less overall number of merges
- Better coverage and time than most similar version of previous work

UNI
BASEL

Thank you!

Questions?