Block-Level Goal Recognition Design

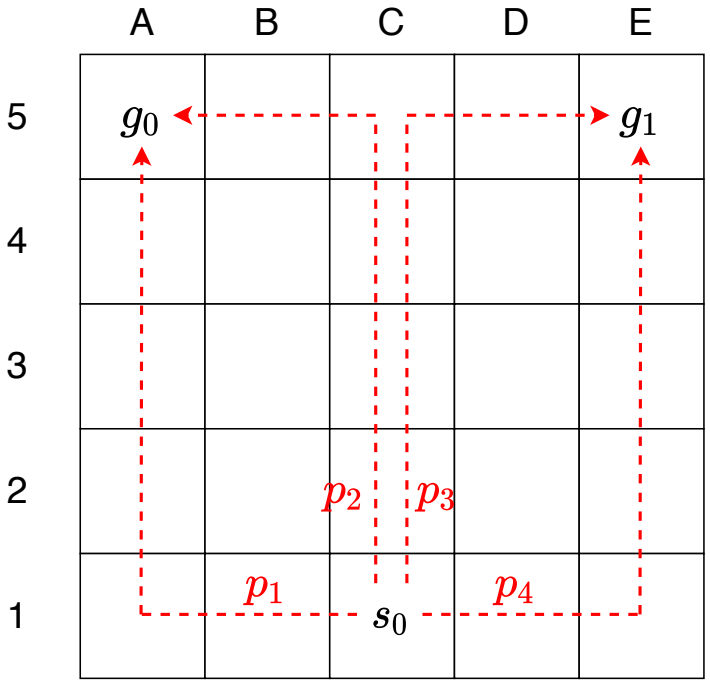
Tsz-Chiu Au

chiu.au@gmail.com

Department of Computer Science and Engineering
Ulsan National Institute of Science and Technology

Goal Recognition Design (GRD)

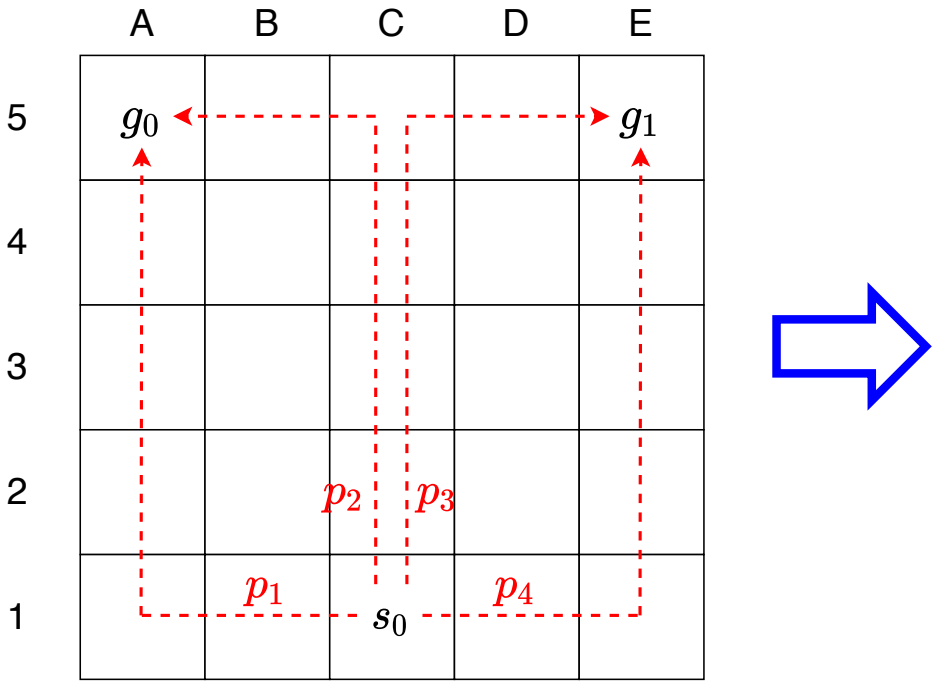
- **Goal recognition** – an observer infers the goal of an agent from a sequence of observations of agents' actions.
- **Goal recognition design**[1] – modify an environment to help observers to recognize the goal of an agent.



[1] Keren et al. Goal Recognition Design. AAAI 2014

Worst Case Distinctiveness (WCD)

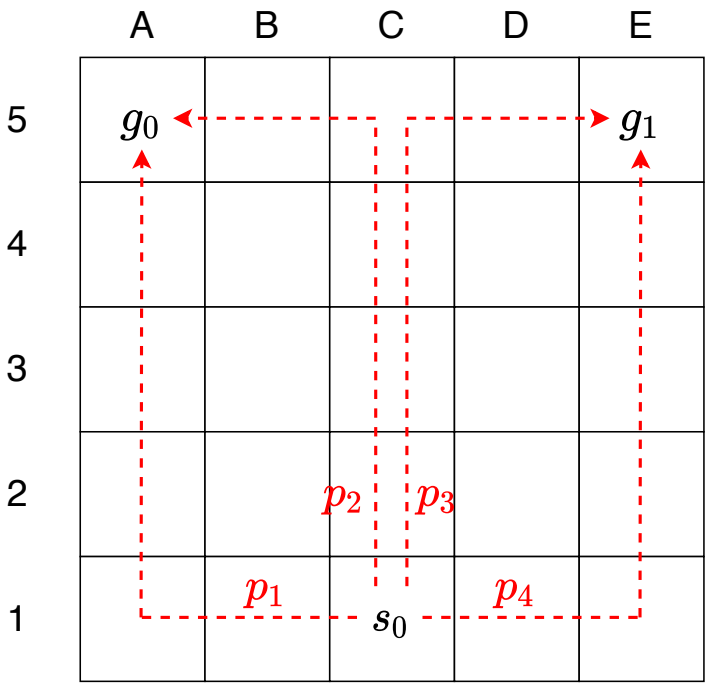
- **Worst case distinctiveness** – a popular objective function for GRD
  - » The highest number of observations that an observer needs to observe *before* it can be certain of the agent's goal in the worst case.



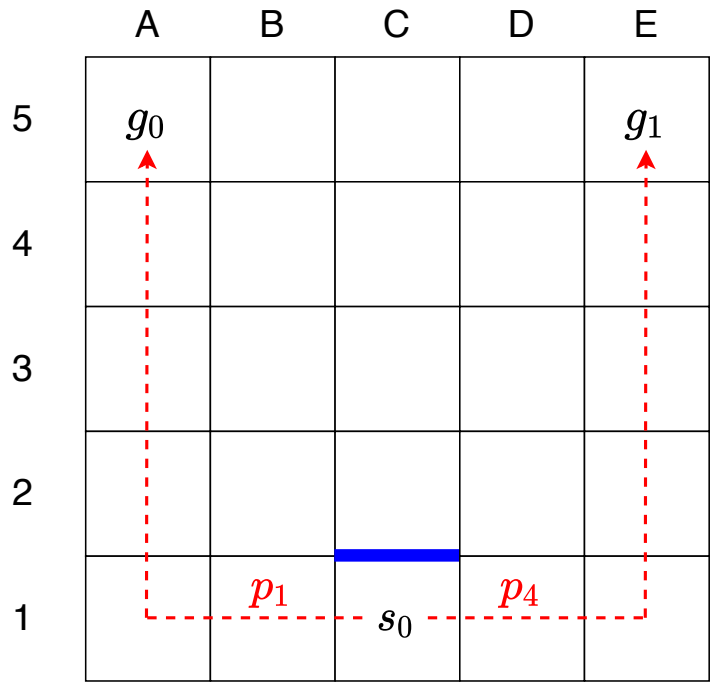
Before redesign, WCD = 4

Minimizing WCD

- GRD aims to find a sequence of modifications to an environment in order to minimize the WCD.



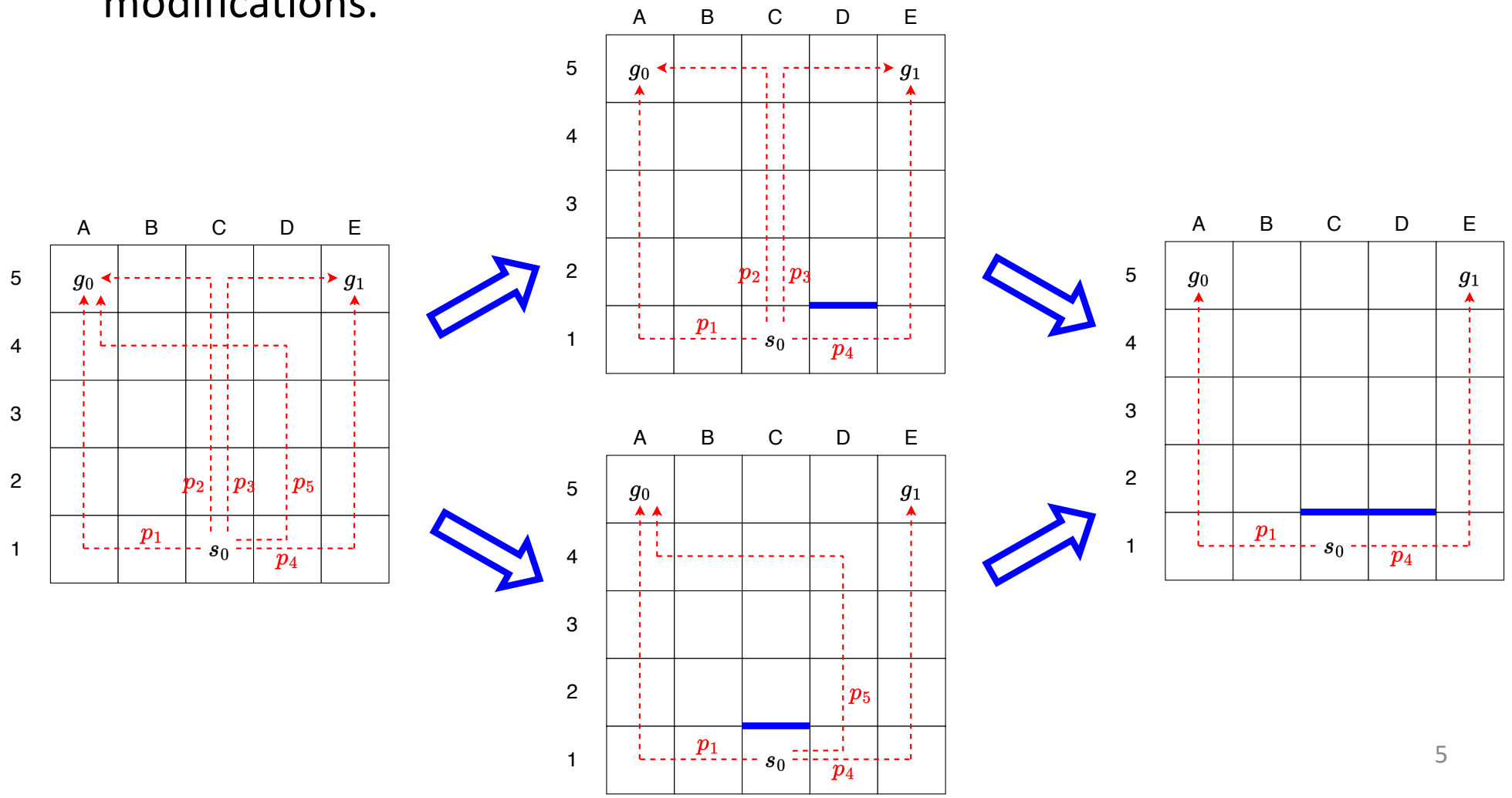
Before redesign, WCD = 4



After redesign, WCD = 0

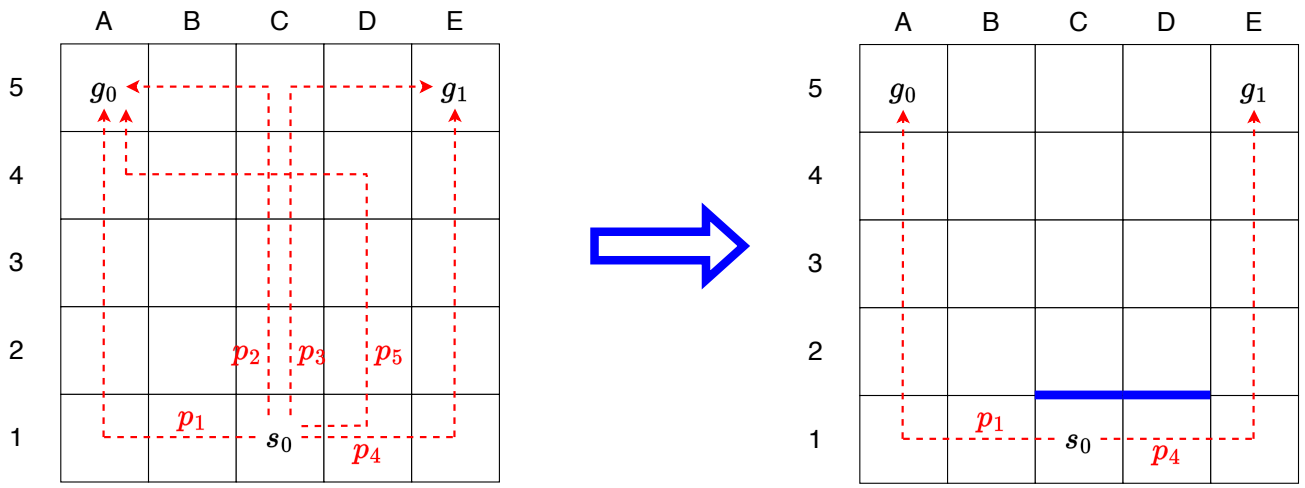
Combinatorial Explosion of Modifications

- We often need more than one modification to modify an environment.
- Existing GRD algorithms will enumerate all combinations of the modifications.



Design Constraints

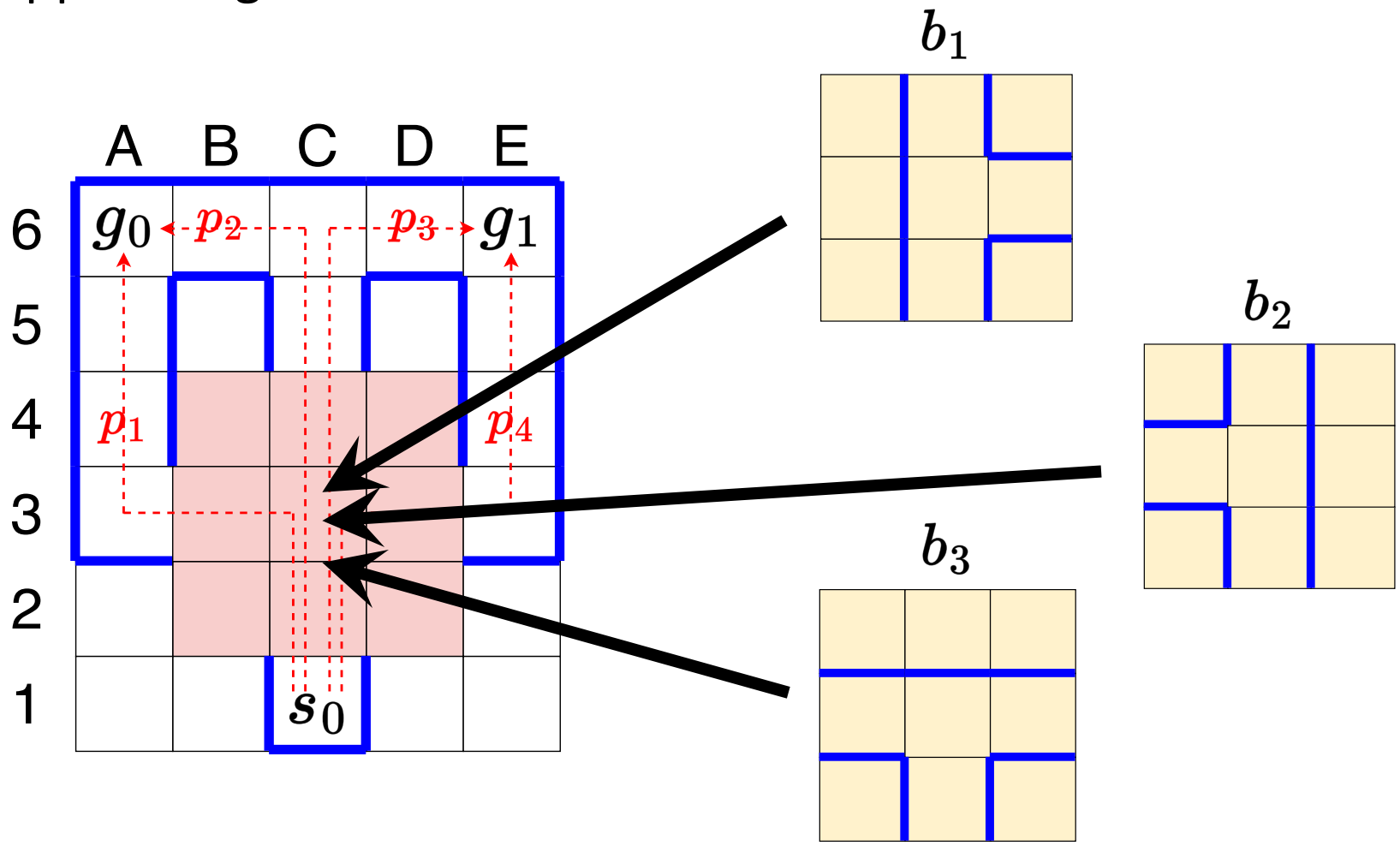
- If the modifications are applied together, we can avoid the enumeration of all possible combinations of modifications.



- In some applications, some modifications must be applied together with respect to the **design constraints**.

Blocks

- A **block** is a group of closely related modifications that are applied together.

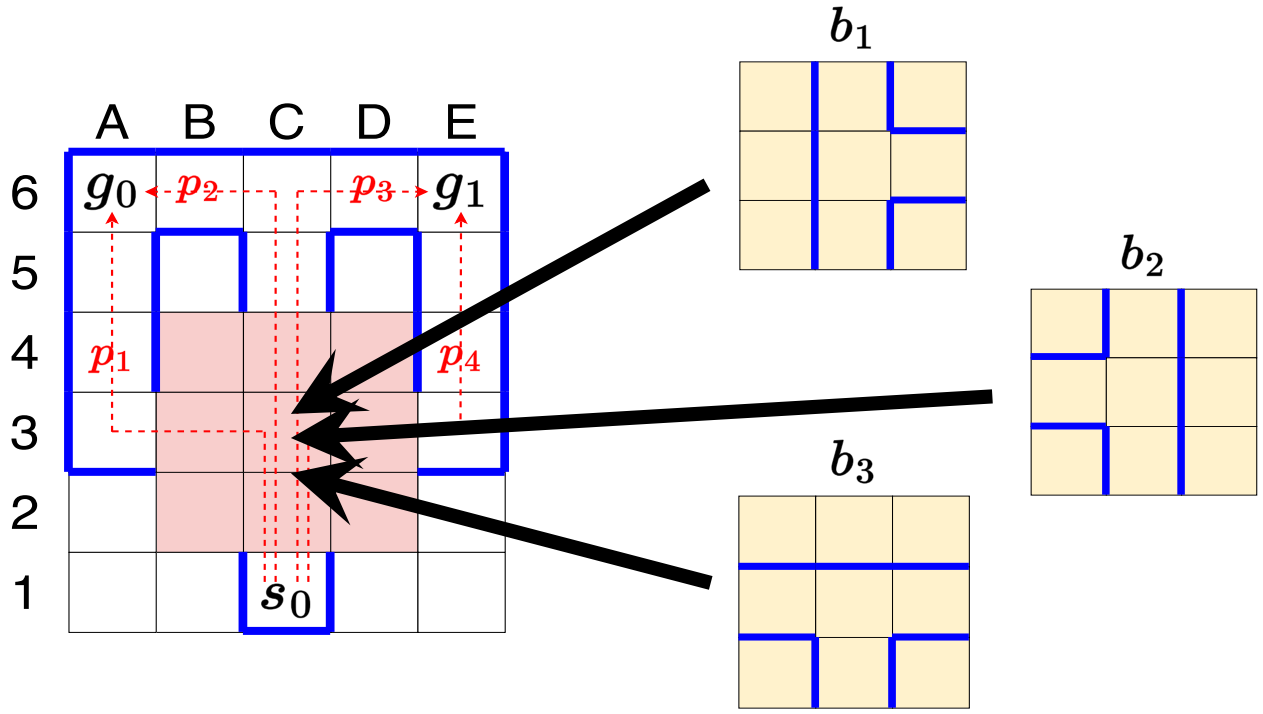


Our Contributions

- We propose replacing existing modifications with **blocks**
  - » which group several closely related modifications together with respect to design constraints.
- **Hierarchical design model**
  - » There could be blocks within blocks
- We devise **new pruning rules** for block-level GRD.
  - » Block-based pruned-reduce
  - » Design subtree pruning rule
- A **local search algorithm** for solving large GRD problems.

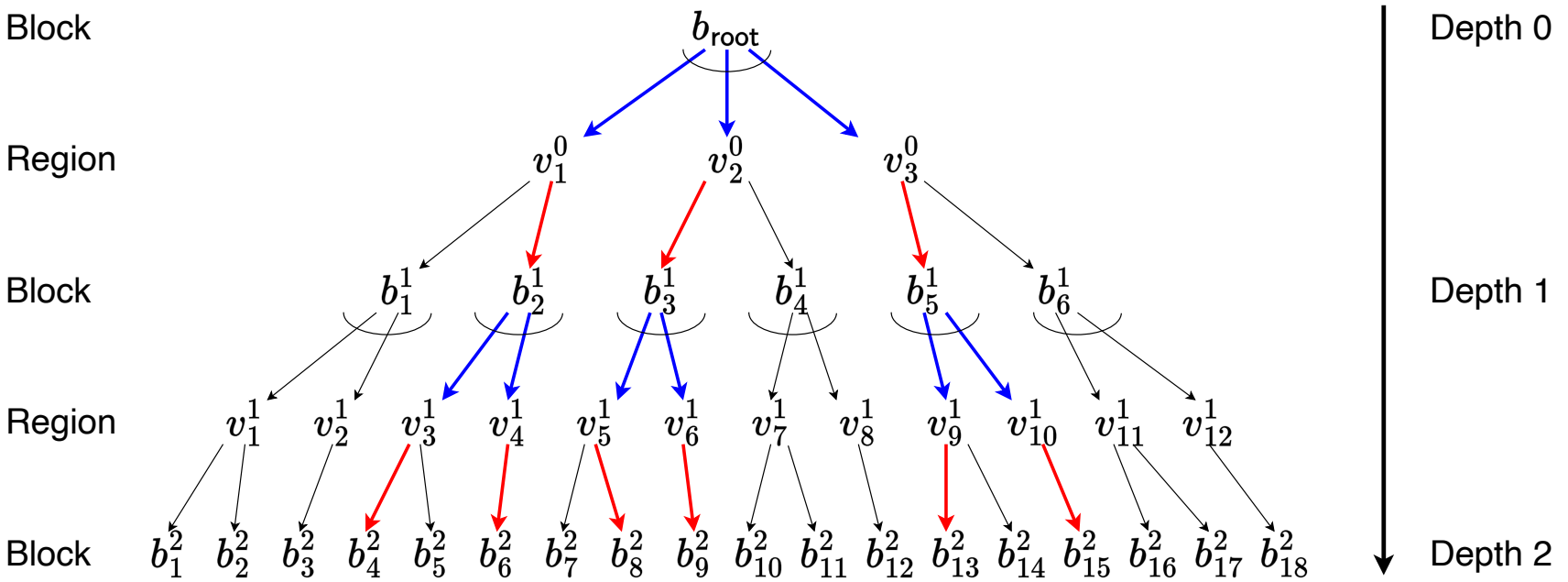
Blocks and Regions

- **Blocks** – a group of related modifications that simultaneously modifies multiple edges in a search space (e.g., b1, b2, and b3)
- **Regions** – a part of a search space that can be substituted by blocks (e.g., the red region).



The Block-Level GRD Problem

- **Hierarchical design model** – each block can have subblocks such that the design space becomes hierarchical.
- The objective is to find a **design tree** (e.g., the red arrows) such that the WCD is minimized for a given set of **legal plans**.

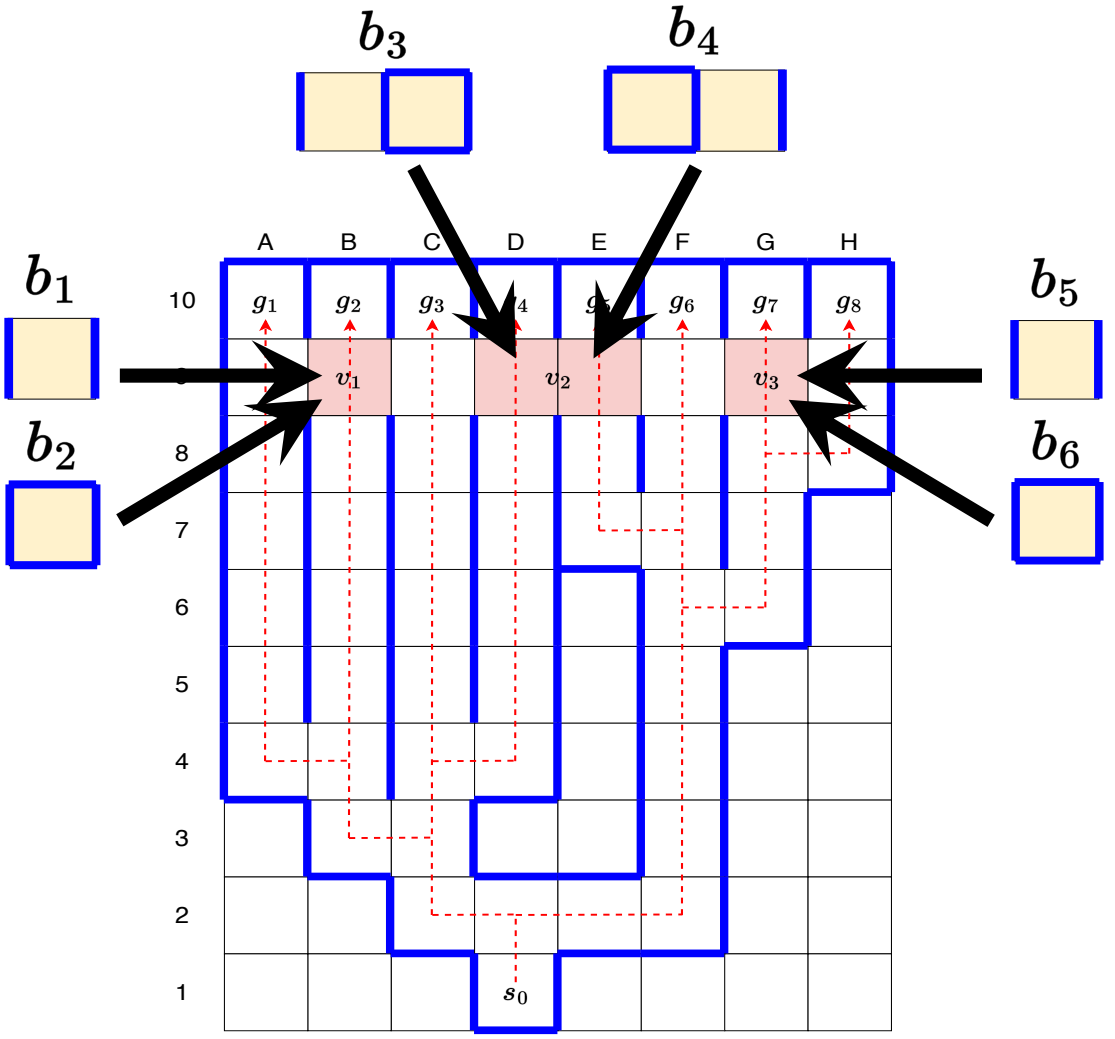


Block-Based Pruned-Reduce

- **Pruned-reduce** – avoid expanding nodes in a design algorithm (e.g., exhaustive-reduce) that cannot reduce the WCD.
- A block b **necessarily invalidates** a legal path p if and only if there exists one subblock b’ in every possible design subtree of b such that b’ invalidates p.
- A design algorithm does not expand b if b does not necessarily invalidate the legal paths that yield the current WCD.
- For better performance, do not expand b if b does not **possibly invalidate** the legal paths that yield the current WCD.

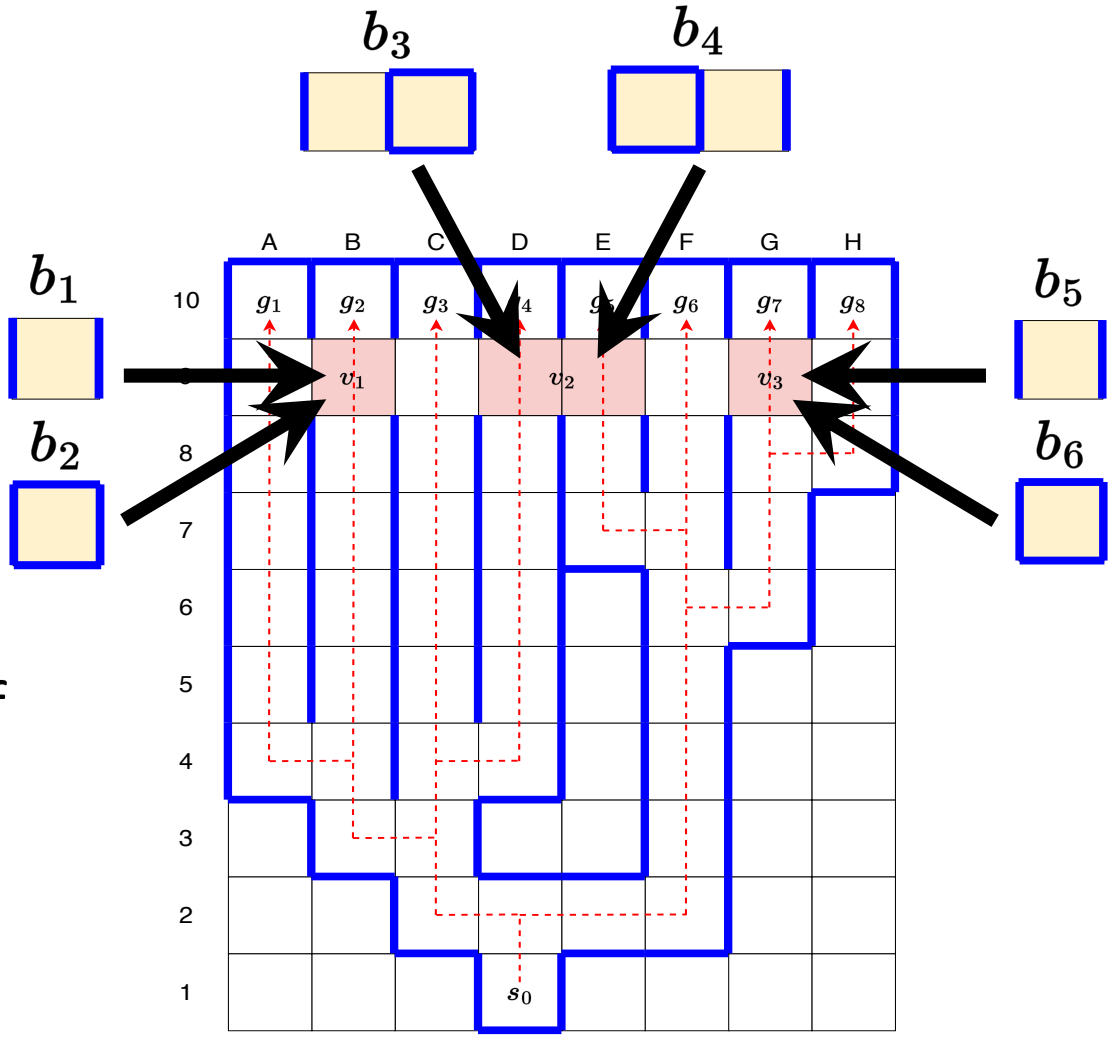
Design Subtree Pruning Rule

- **Pruning design subtrees** – avoid expanding some design subtrees that cannot reduce the WCD.



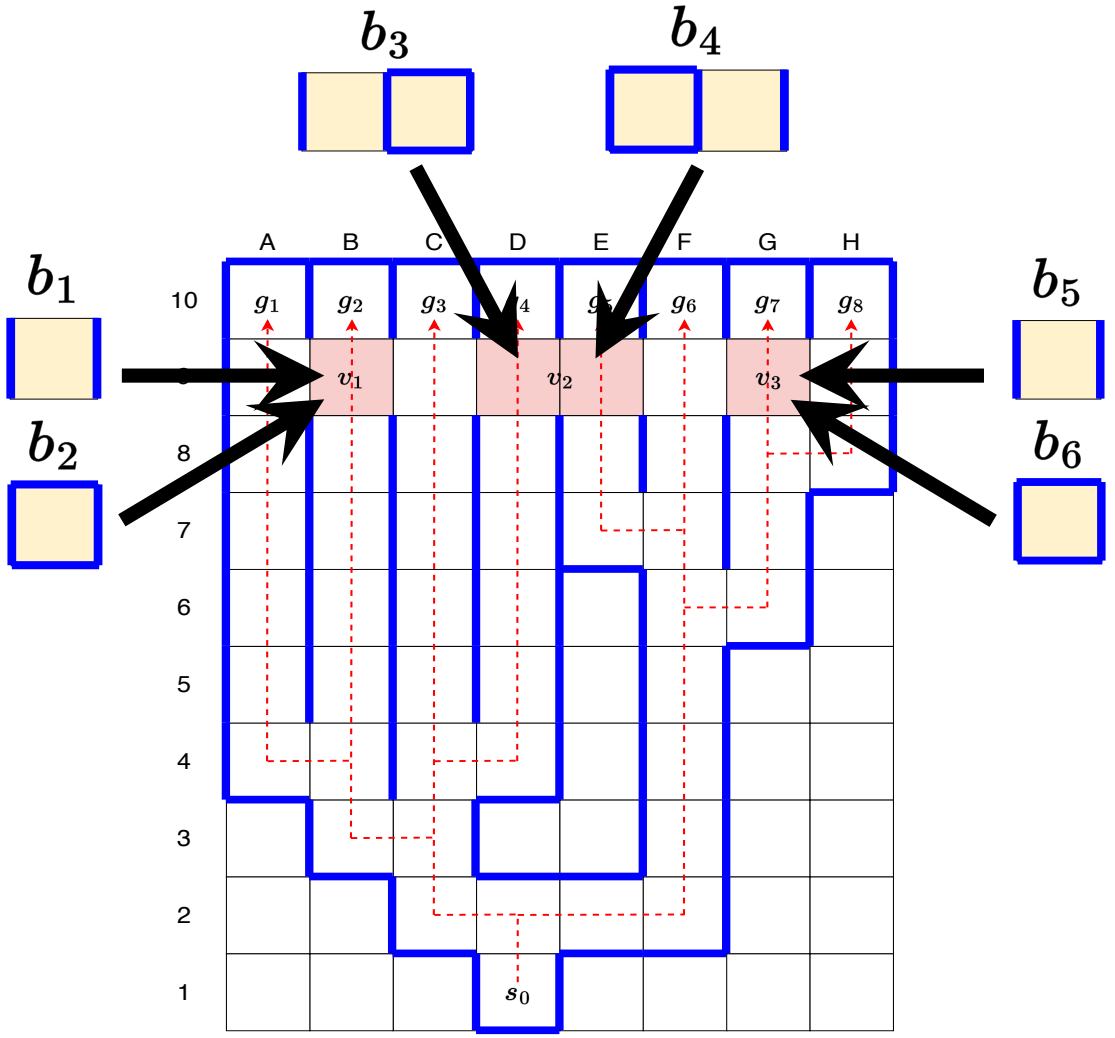
Design Subtree Pruning Rule (cont.)

- Compute the lower and upper bounds of the **relative WCD** of every vertex in a legal path tree.
- For every junction in a legal path tree,
  - » if the lower bound of a child vertex is larger than the upper bound of another child vertex,
    - mark all regional vertices in the subtrees as “pruned”.



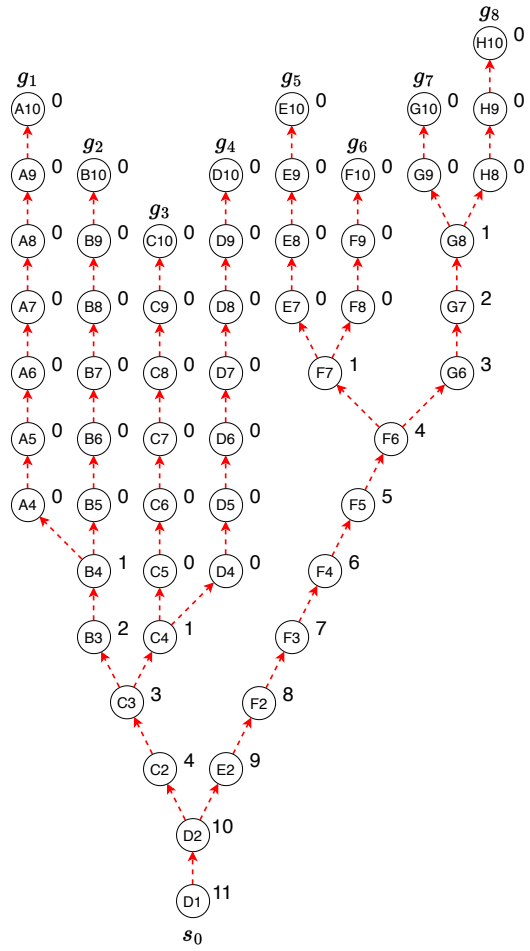
Design Subtree Pruning Rule (cont.)

- If all instances of a regional vertex are marked as "pruned", we can prune its design subtrees.

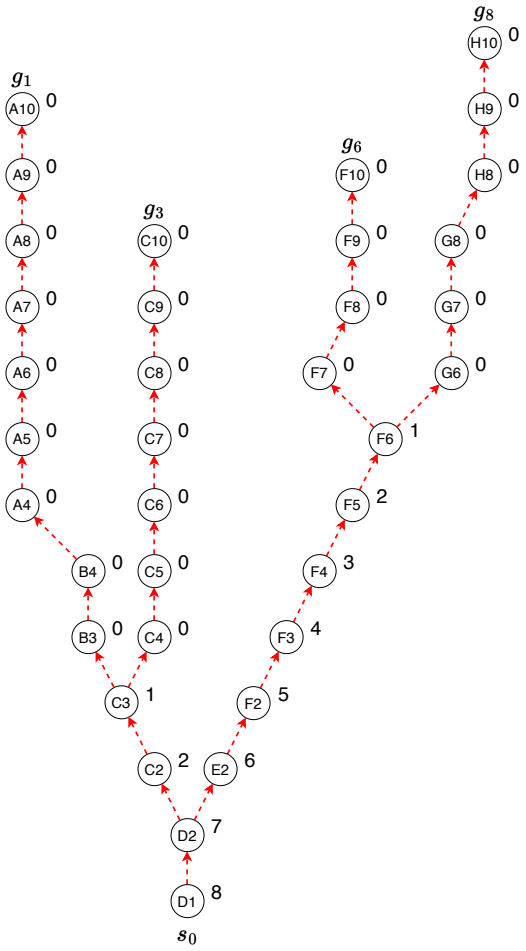


Design Subtree Pruning Rule (cont.)

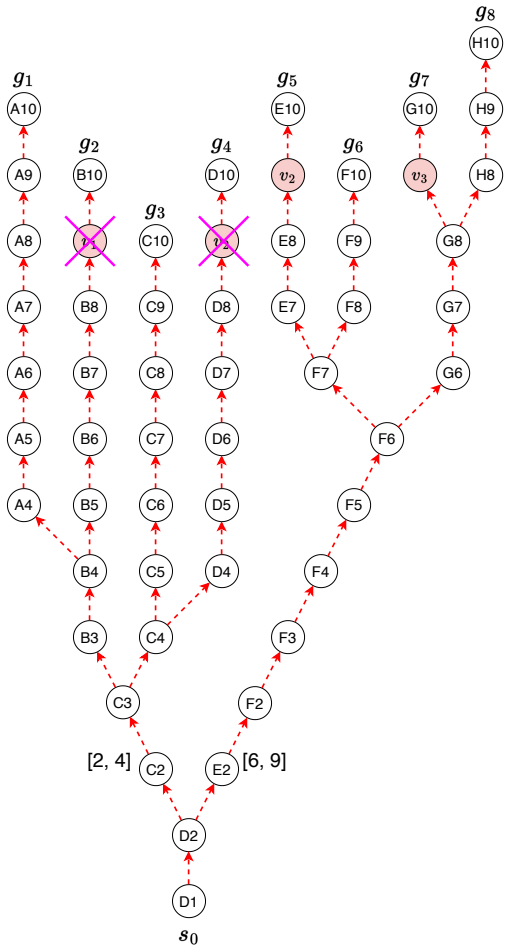
Legal path tree for computing the upper bounds: T



Legal path tree for computing the lower bounds: $T^{\min}$



Compact Path Tree: T^{compact}



Empirical Evaluation

	Original BFS		Our BFS		Local Search	
	Time	WCD	Time	WCD	Time	WCD
LOGISTICS	23.67	8.0	2.48	8.0	0.43	8.5
DEPOTS	17.29	4.2	0.80	4.2	0.24	4.2
GRID	37.64	8.5	3.24	8.5	0.52	8.6
DRIVERLOG	16.80	7.7	0.77	7.7	0.15	7.9

Table 1: The average execution times of the BFS (in second) and the average minimum WCDs in Experiment 1.

- Our hierarchical design model substantially reduced the size of the search space.
- The local search algorithm outperformed the BFS, whereas the average minimum WCDs were not much larger.

Empirical Evaluation (cont.)

	No Pruning	Pruned-reduce	P.R. + D.S.P.
LOGISTICS	20.80	2.61	2.48
DEPOTS	5.25	0.89	0.80
GRID	33.02	3.55	3.24
DRIVERLOG	5.17	0.82	0.77

Table 2: The average execution times of the algorithms (in second) in Experiment 2.

- Both pruning rules can speed up the BFS.
  - » But design subtree pruning was less effective than pruned-reduce
- The performance of design subtree pruning depends on
  - » How many design subtrees are pruned
  - » How large the pruned design subtrees are

Summary and Future Work

- We can enforce some design constraints among modifications by using blocks in deterministic GRD.
- Our experiments showed that
  - » Block-based pruned-reduce is highly effective.
  - » Design subtree pruning is effective when many large design subtrees are pruned.
- Despite its name, the design subtree pruning rule is also applicable to non-hierarchical design models.
- In the future
  - » Combine our hierarchical design model with hierarchical legal plans
  - » Extend our model with sensing actions for partial observability.

Thank you!