Fast Maximum Common Subgraph Search: A Redundancy-Reduced Backtracking Approach

Kaiqiang Yu[1] Kaixin Wang[2] Cheng Long[1] Laks Lakshmanan[3] Reynold Cheng[4]

[1]Nanyang Technological University, Singapore; [2]Beijing University of Technology, China; [3]The University of British Columbia, Canada; [4]The University of Hong Kong, China

Background: Maximum Common Subgraph Search

Graph Isomorphism

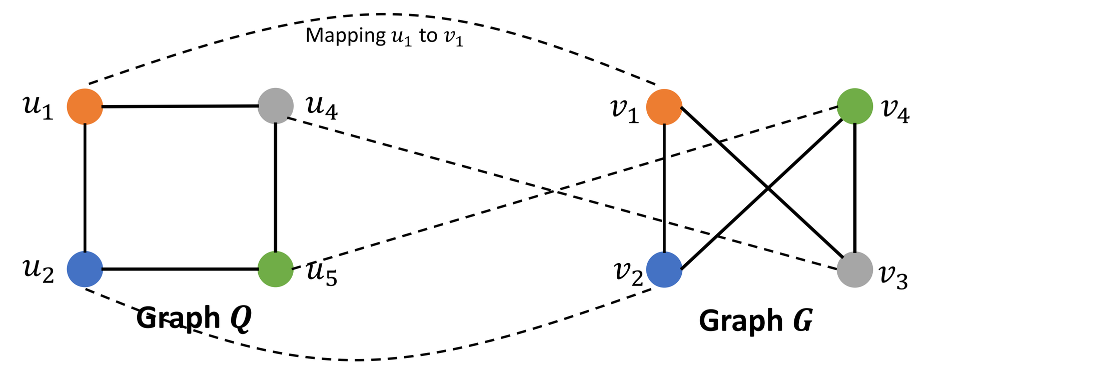

Figure 1. Q is isomorphic to G under bijection $\phi: u_1 \to v_1,\ u_2 \to v_2,\ u_3 \to v_4,\ u_4 \to v_3$

Common Induced Subgraph

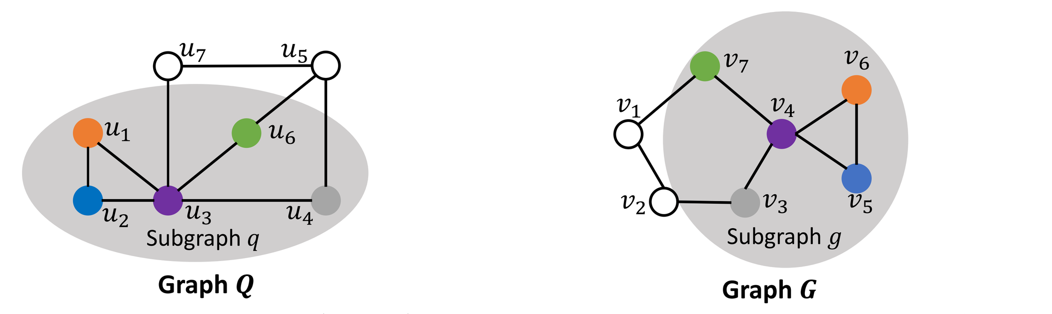

Figure 2. Common subgraph $\langle q, g, \phi \rangle$ where q is *isomorphic* to g under bijection ϕ

Maximum Common Subgraph (MCS) Search

Input: two graphs $Q = (V_Q, E_Q)$ and $G = (V_G, E_G)$.

Output: the common subgraph of Q and G with the largest number of vertices.

Applications: Drug discovery, cheminformatics and etc.

State-of-the-art backtracking method: McSplit [2, 1]

Main idea: recursively expand a partial solution via *backtracking* process.

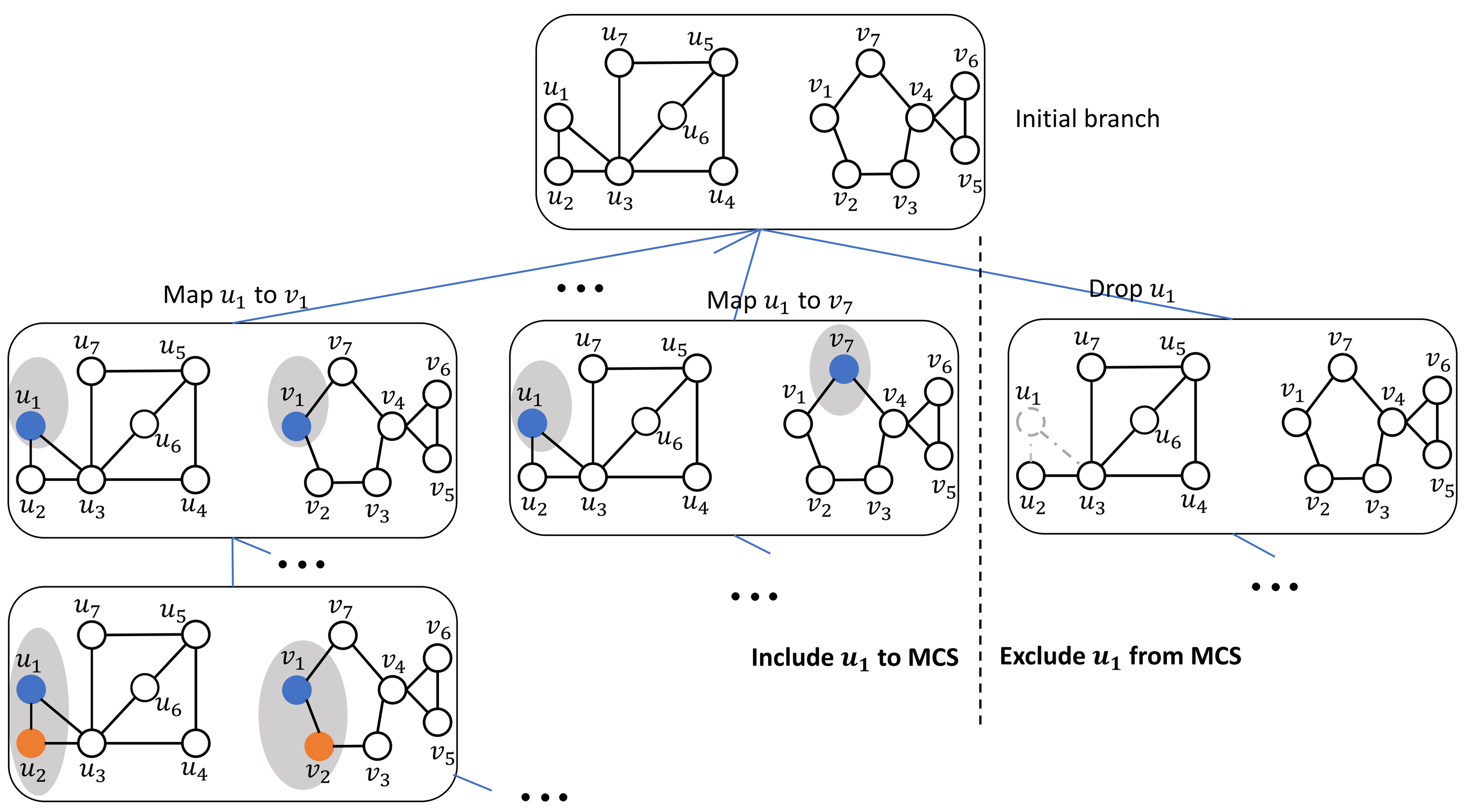

Figure 3. Illustrating the backtracking process

Our Algorithm: RRSplit

`RRSplit` follows the above backtracking process and adopts two types of reduction methods, namely *vertex-equivalence-based reduction* and *maximality-based reduction*, for saving computations.

I. Vertex-Equivalence-based Reduction

Motivation: redundancies induced by *common subgraph (cs) isomorphism*

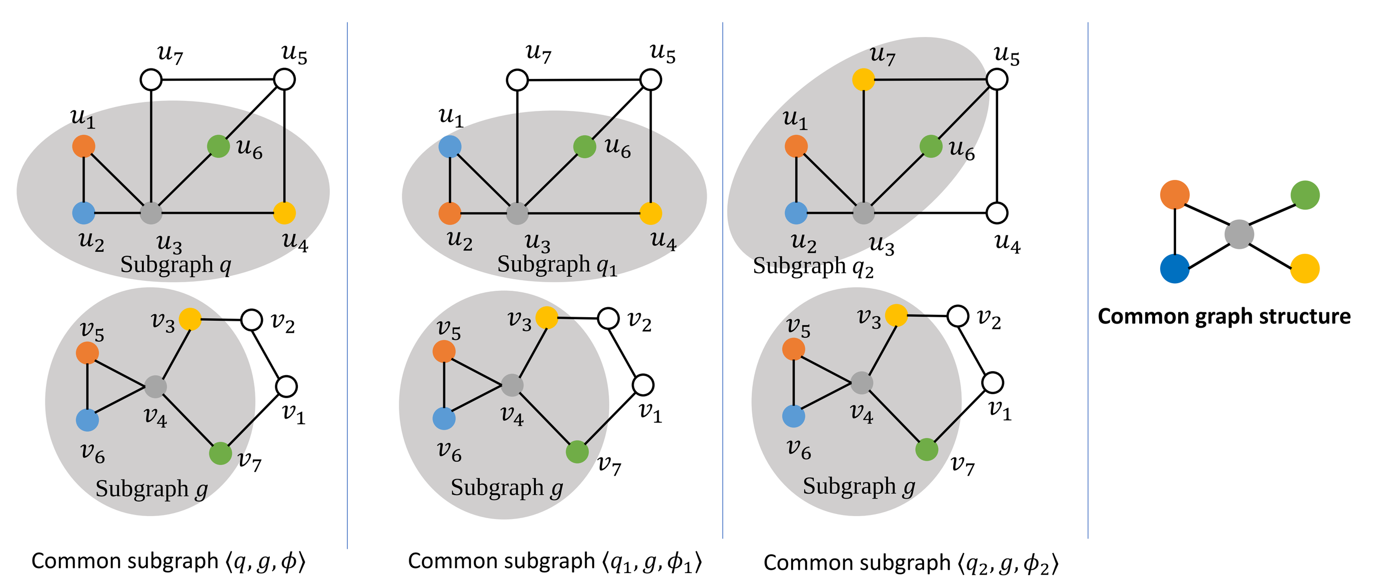

Figure 4. $\langle q, g, \phi \rangle$, $\langle q_1, g, \phi_1 \rangle$ and $\langle q_2, g, \phi_2 \rangle$ are mutually cs-isomorphic since they have the *same topological structure*. *Exploring them via backtracking is redundant.*

Idea: reduce the redundancy by *vertex-equivalence-based reduction*

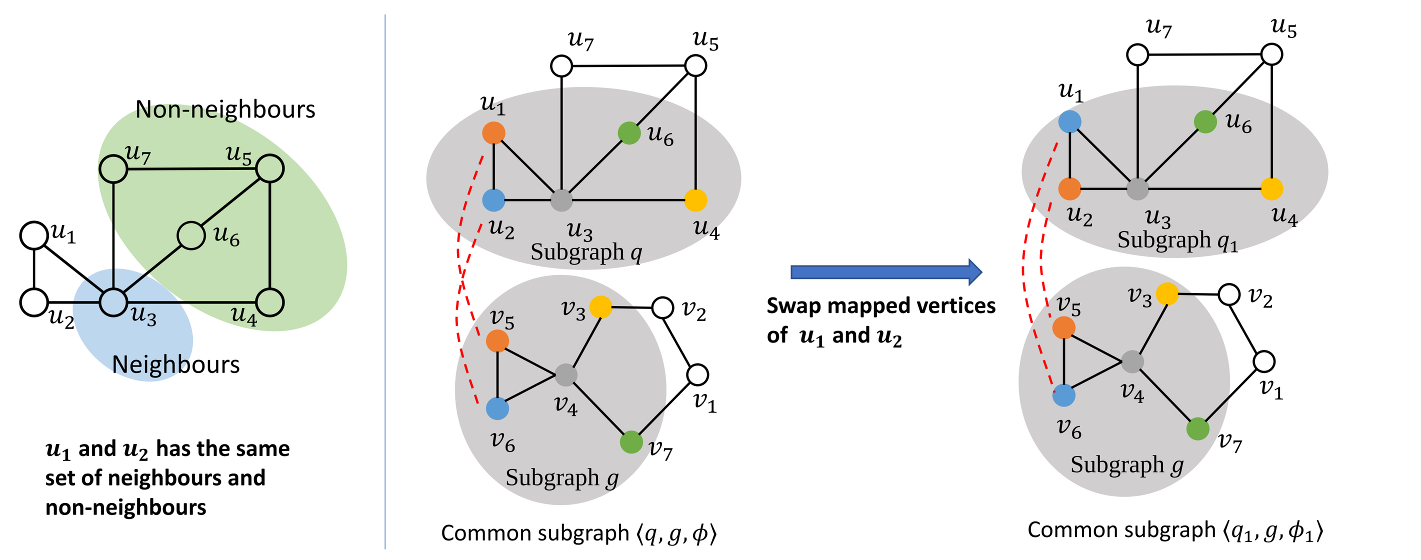

Figure 5. Identify cs-isomorphic common subgraphs based on two structurally equivalent vertices u_1 and u_2

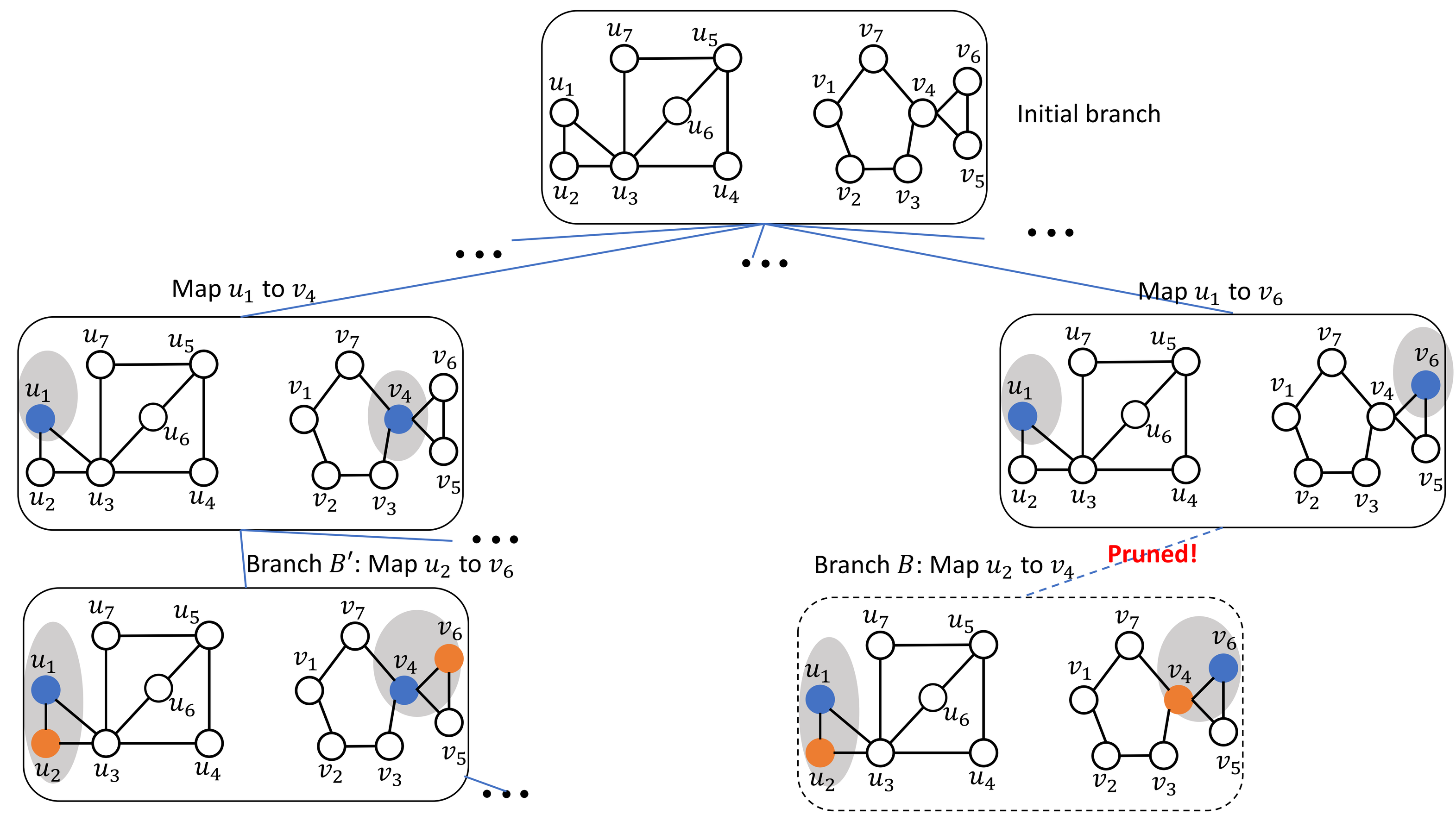

Figure 6. Illustrating the reduction rule. Branch B can be pruned since we prove that *all MCSs found within B are cs-isomorphic to those found previously in B'* (given u_1 and u_2 are structurally equivalent)

For details of other vertex-equivalence-based reduction rules, please refer to our paper.

II. Maximality-based Reduction

Main idea.

- A maximum common subgraph must be a maximal common subgraph
- Prune those branches that hold non-maximal common subgraphs only

III. Summary and Analysis

Assume $|V_G| \geq |V_Q|$:

- Space complexity: $O(|V_Q| + |S^*| \times |V_G|)$ (S^* - the MCS)
- Time complexity: $O^*((|V_G| + 1)^{|V_Q|})$ (O^* ignores the polynomials; We remark that *the SOTA McSplitDAL [1] runs fast in practice but does not have any theoretical guarantees*)

Experiments

Dataset: biochemicalReactions (BI), images-CVIU11 (CV), images-PR15 (PR) and LV (LV)

Baselines: the state-of-the-art `McSplitDAL` [1]

Table 1. Datasets used in the experiments ("# of solved instances" refers to the number of instances solved by algorithms within 1,800 seconds and "Achieved speedups" refers to the percentage of the solved instances that `RRSplit` runs at least 5×/10×/100× faster than `McSplitDAL`)

Dataset	Domain	# of graphs	# of instances	# of vertices	# of solved instances		Achieved speedups		
					`RRSplit`	`McSplitDAL`	5×	10×	100×
BI	Biochemical	136	9,180	9∼ 386	7,730	4,696	91.3%	84.4%	69.7%
CV	Segmented images	190	6,424	22∼ 5,972	1,351	1,291	76.5%	48.6%	0.2%
PR	Segmented images	25	24	4∼ 4,838	24	24	91.7%	91.7%	58.3%
LV	Synthetic	112	6,216	10∼ 6,671	1,059	883	68.0%	54.7%	38.3%

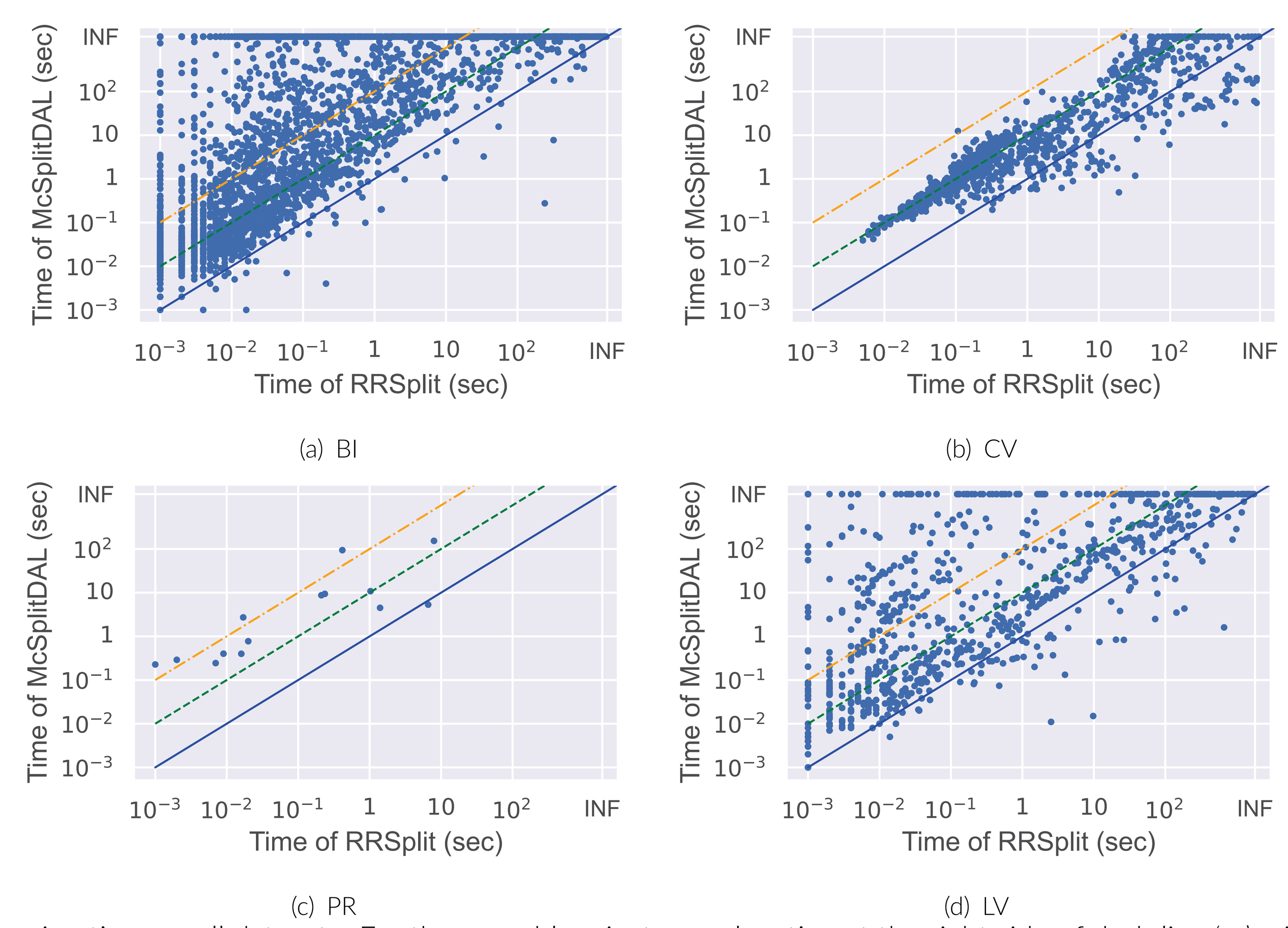

Figure 7. Running time on all datasets. For those problem instances locating at the right side of dash line '- -' with orange color (resp. '- -' with green color), `RRSplit` achieves at least 100× (resp. 10×) speedup compared with `McSplitDAL`.

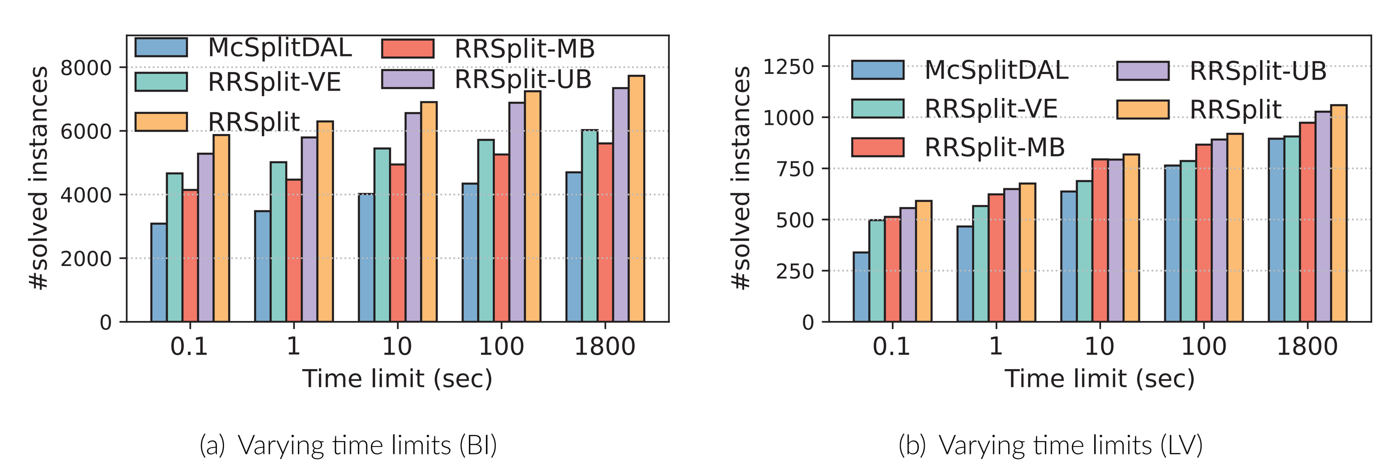

Figure 8. Ablation study: comparison among various reductions (RRSplit-VE: without vertex-equivalence-based reduction, RRSplit-MR: without maximality-based reduction, RRSplit-UB: without the proposed upper bound)

References

[1] Yanli Liu, Jiming Zhao, Chu-Min Li, Hua Jiang, and Kun He. Hybrid learning with new value function for the maximum common induced subgraph problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 4044–4051, 2023.

[2] Ciaran McCreesh, Patrick Prosser, and James Trimble. A partitioning algorithm for maximum common subgraph problems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, pages 712–719, 2017.