Comparative Analysis of DSA, DSAN, and MGM Algorithms

Dorian Benhamou Goldfajn

October 8, 2024

1 Introduction

This report analyzes the impact of various parameters on the performance of three algorithms: DSA (Distributed Stochastic Algorithm), DSAN (Distributed Simulated Annealing), and MGM (Maximum Gain Message). We'll examine how each parameter affects the average cost and the ability to minimize it for each algorithm.

2 Experimental Setup

2.1 DCOP Benchmark Generation

Each DCOP instance is generated as a graph with varying numbers of agents (nodes), edge densities, and graph structures (e.g., complete, Erdos-Renyi). Cost Tables are assigned to edges, representing the constraints between agents.

2.2 Algorithm Parameters

The algorithms are executed for 200 iterations over 10 randomly generated instances for each combination of parameters. The key parameters varied include cost range, edge density, graph type, number of actions, number of agents, DSA thresholds, DSAN temperature, and number of iterations.

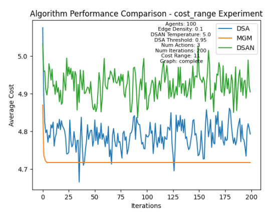
3 Parameter Effects

3.1 Cost Range

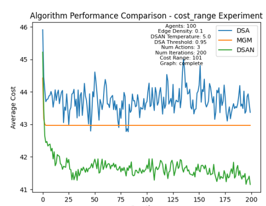
Effect on average cost:
It comes as no surprise that wider cost ranges consistently lead to higher average costs across the graphs (regardless of algorithm).
Effect on algorithm performance:

- **DSA:** Performance may relatively slightly degrade with very wide cost ranges due to increased fluctuations, although performance stays mostly stable.
- **DSAN:** Generally maintains good performance across various cost ranges and seems to improve with wider cost ranges. Performs particularly bad for really low range.
- **MGM:** Relative to the other algorithms, performs worse as cost ranges increase. I believe it struggles with very wide ranges due to a quick local minima.

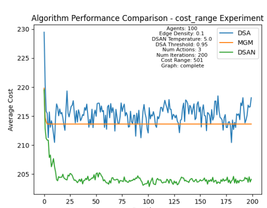
Recommendation: DSAN is likely the best choice for scenarios with wide or unpredictable cost ranges. DSA and MGM may fit better for scenarios with low costs.



(a) Cost Range: 1-10



(b) Cost Range: 1-100



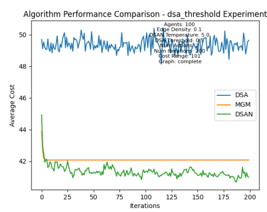
(c) Cost Range: 1-500

Figure 1: Performance across different cost ranges.

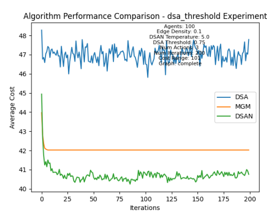
3.2 DSA Threshold

Effect on DSA Performance: As expected, higher thresholds lead to generally better results. However, high thresholds are more prone to get trapped at a local minima. Therefore it's important to find a balanced threshold which prioritizes the 'best' action and minimizes local minima traps.

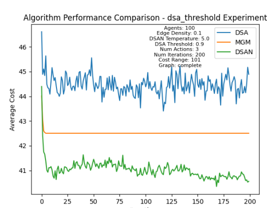
Recommendation: For DSA, tune this parameter based on the specific problem. Start with a moderate threshold (e.g., 0.9) and adjust based on performance.



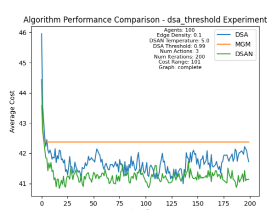
(a) threshold = 0.5



(b) threshold = 0.75



(c) threshold = 0.9



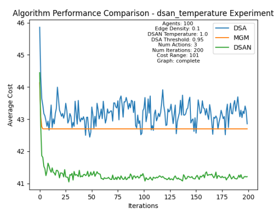
(d) threshold = 0.99

Figure 2: Performance by DSA threshold parameter.

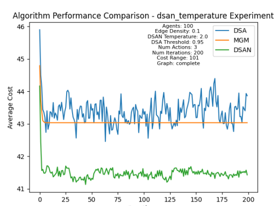
3.3 DSAN Temperature

Effect on DSAN performance: Performance tends to remain well across all temperatures. However, higher temperatures increase exploration, and consequently, the graph's deviation which may lead to sub-optimal solutions.

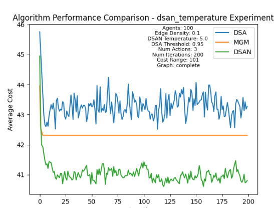
Recommendation: Similarly to DSA's threshold, tune this parameter based on the specific problem. Start with a moderate threshold (e.g. 2) and adjust based on performance. Another approach could be to start with a high temperature for exploration and gradually decrease it for exploitation.



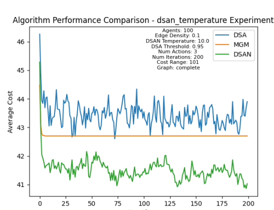
(a) temp = 1



(b) temp = 2



(c) temp = 5



(d) temp = 10

Figure 3: Performance by DSAN temperature parameter.

3.4 Edge Density

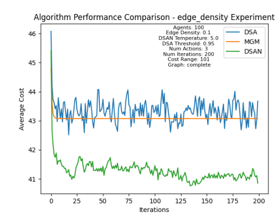
Effect on average cost:

Higher edge density naturally leads to higher average costs due to increased constraints, making it more complex to compare performance.

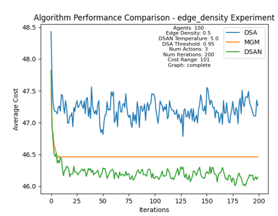
Effect on cost minimization:

- **DSA:** Performance may degrade with very high edge densities due to increased local optima.
- **DSAN:** Generally robust to changes in edge density.
- **MGM:** Can perform well with high edge densities due to its coordination mechanism.

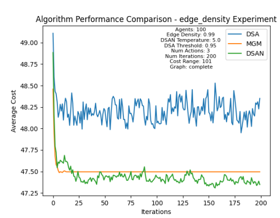
Recommendation: MGM or DSAN for high edge density scenarios, DSAN otherwise.



(a) edge density = 0.1



(b) edge density = 0.5



(c) edge density = 0.99

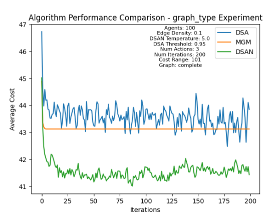
Figure 4: Performance by edge density.

3.5 Graph Type

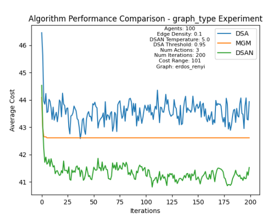
Effect By Graph Type:

- **Complete:** DSAN performs best, MGM performs slightly better than DSA. DSA seems to slightly improve with more iterations.
- **Erdos Renyi:** Not so different than the complete graph. DSAN performs best and MGM performs better than DSA by a noticeable amount. DSA may perform slightly worse.
- **Barbasi Albert:** Average costs appear lower than in the previous two graphs. MGM does not work on this graph and immediately converges. DSAN outperforms DSA while both algorithms appear to extremely deviate and fluctuate, indicating a tendency to explore in this type of graph.

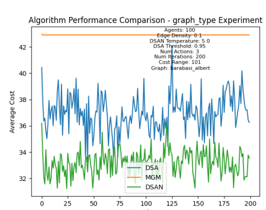
Recommendation: DSAN is the safest choice when graph structure is unknown. MGM may outperform DSA but could get trapped in graphs like Barbasi Albert.



(a) complete graph



(b) Erdos Renyi graph



(c) Barbasi Albert Graph

Figure 5: Performance by graph.

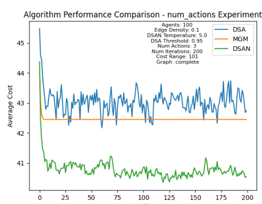
3.6 Number of Actions

Effect on average cost: No change as each agent still picks a single action.

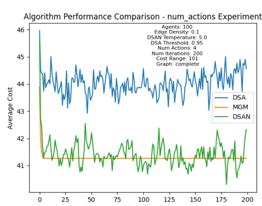
Effect on Algorithm Performance:

- **DSA:** Less noisy (deviation) with more action choices. Performs on par with MGM with low actions, but under-performs with more action choices. DSA always under-performs DSAN.
- **DSAN:** Scales somewhat well with increasing number of actions due to its learning approach. Performs worse than MGM with higher number of action choices while outperforming DSA.
- **MGM:** Performs particularly well for higher number of action choices while performing worse than DSAN for lower number of action choices.

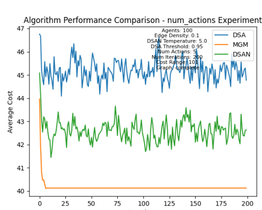
Recommendation: DSAN for unknown or low-moderate number of action choices. MGM for high amount of action choices.



(a) 3 actions



(b) 4 actions



(c) 5 actions

Figure 6: Performance by Number of Action Choices

3.7 Number of Agents

Effect on average cost: On average, costs stay the same despite higher number of agents.

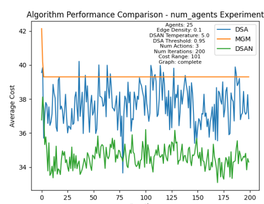
Effect on Algorithm performance:

- **DSA:** Scales reasonably well but may struggle with large numbers of agents. Typically performs better than MGM in low-medium amount of agents, but under performs DSAN across low or high number of agents.

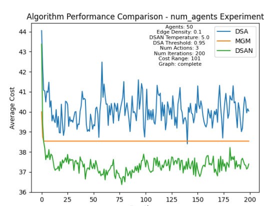
- **DSAN:** Good scalability. Out performs both DSA and MGM across any number of agents.

- **MGM:** Works better than DSA for medium-high amount of agents, but worse than DSAN and DSA otherwise.

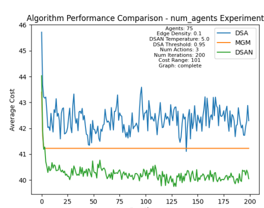
Recommendation: DSAN for medium-large-scale problems with many agents. DSA or DSAN may both work for smaller problems.



(a) 25 agents



(b) 50 agents



(c) 75 agents

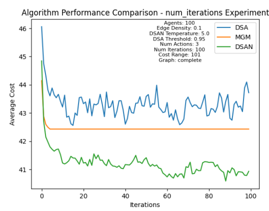
Figure 7: Performance by number of agents.

3.8 Number of Iterations

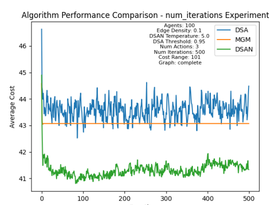
Effect on Algorithm performances:

- **All algorithms:** Generally, more iterations lead to better performances. MGM tends to converge early. DSAN consistently outperforms the rest while DSA under performs and typically gets stuck in a "bouncing zone"

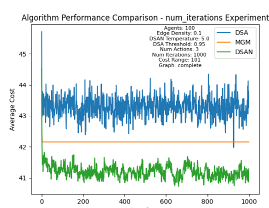
Recommendation: Start with a moderate number of iterations and increase if needed.



(a) 100 iterations



(b) 500 iterations



(c) 1000 iterations

Figure 8: Performance by number of iterations.

---

4 Interesting Patterns and Correlations

4.1 Exploration-Exploitation Trade-off

DSAN's performance is highly correlated with its temperature parameter, showcasing the classic exploration-exploitation dilemma in reinforcement learning. Higher temperatures lead to more exploration early on, while lower temperatures result in faster convergence.

4.2 Scalability vs. Optimality

As the problem size increases (more agents, actions, or higher edge density), there's often a trade-off between scalability and finding the optimal solution. DSAN and MGM tend to handle this trade-off better than DSA.

4.3 Graph Structure Impact

The performance of all algorithms, especially DSA, is correlated with the graph structure. More complex structures often lead to reduced performance, with DSAN showing the most resilience to this effect.

4.4 Convergence Speed

MGM often converges faster than DSAN or DSA for most problems, but DSAN tends to find better solutions given enough iterations, especially for complex problems. DSA typically finds a fair bouncing zone fast.

5 Observations and Explanations

5.1 DSA's Sensitivity to Initial Conditions

DSA's performance can vary significantly based on initial conditions, especially in complex scenarios. This is due to its stochastic nature and potential for getting stuck in local optima.

5.2 DSAN's Learning Curve

DSAN might perform poorly in early iterations but significantly improve over time. This is due to its learning mechanism adapting to the problem structure.

5.3 MGM's Consistency

MGM often shows more consistent performance across different problem instances of similar complexity. This is due to its deterministic nature and focus on maximum gain.

5.4 Trivial Solutions in Low Density Graphs

In very low-density graphs, all algorithms might find optimal or near-optimal solutions quickly. This is because the problem becomes closer to independent optimization for each agent, with fewer inter-agent constraints to complicate the solution space.

5.5 Diminishing Returns on Iterations

All algorithms show diminishing returns with increased iterations, but the point at which this occurs varies. DSAN typically benefits from more iterations compared to DSA and MGM, as it takes time for the learning process to converge to better solutions. However, after a certain number of iterations, further improvements become minimal.

6 Conclusion

The choice of the best algorithm depends significantly on the specific characteristics of the problem at hand. Some general guidelines emerge from the analysis:

- **For simple, small-scale problems with clear structure**, DSA can be effective and computationally efficient.
- **For complex, large-scale problems with varied or unknown structures**, DSAN often provides the best results given sufficient learning time.
- **For problems with clear local structures and moderate complexity**, MGM offers a good balance of performance and consistency.

Future work could involve hybrid approaches that combine the strengths of these algorithms or adaptive methods that switch between algorithms based on problem characteristics. Additional investigation into parameter tuning strategies, such as dynamically adjusting thresholds or temperatures, could further enhance performance.