3. EFFECTIVE PROBABILITY CALCULATION

In order to effectively calculate the probability of a fault in the system, we suggest a different algorithm, based on the s-combs policy definition. First, we will provide a way to calculate the probability of a single s-comb policy.

3.1 Single S-Comb Calculation

Assume a system with a policy φ composed of a single s-comb, C. At run-time, an agent gets an *obscured observation* matrix W, representing the probability of each agent to be in each state at that time. The policy allows each agent a_i to be in *any* of the states marked '1' in row C_i. Let matrix H be the result of an element-wise product of C and W: $H = C \wedge W$. In this matrix, each element h_{ij} represents the probability of agent a_i to be in state s_j if this state is legal, or 0 if it is not. The probability of agent a_i to be in *some* legal state is therefore the summary of all the elements in row H_i. The product of all agents' probabilities provides the overall probability that the system has no fault:

$$\prod_{i=1}^{n} \sum_{j=1}^{m} h_{ij} \tag{2}$$

This calculation takes only $O(nm)$ time and space.

3.2 Multi S-Comb Calculation

Calculation of a multi s-comb policy must consider combinations that are defined by more than one s-comb in the policy rule, making them non-independent. In probability theory, the probability of several non-independent events' union is given by the inclusion-exclusion principle [2]. In our case, 'events' are s-combs, and we get:

$$P\left(\bigcup_{k=1}^{\ell} R^k\right) = \sum_{k=1}^{\ell}\left[(-1)^{k-1} \sum_{\substack{C\subseteq\{1,\ldots,\ell\}\\ |C|=k}} \left(\prod_{i=1}^{n}\sum_{j=1}^{m} H_{ij}^{C}\right)\right] \tag{3}$$

where H^C is the matrix $W \wedge \bigwedge_{c\in C} R^c$ (element-wise product of the ijth elements of the matrix W and all the matrices R^c). The time complexity of this calculation is linear in the number of agents and states, but is exponential in the number of s-combs in the policy—$O(nm\ell 2^{\ell})$; its space complexity is linear in all three parameters.

Fig. 1 shows the empiric calculation time of identical systems using the Naïve Algorithm vs. our S-Comb one. For the same number of s-combs, increase in the number of agents or states results in minor linear change of the S-Combs run time, but in exponential growth in the Naïve run time. Having more '1' elements in the policy s-combs ('density', [5]) results in more combinations defined by the same ℓ matrices of $n \times m$. That changes nothing in the S-Comb run time, but highly increases the Naïve run time. Only increasing ℓ affects the S-Comb curve mush more than the Naïve. In fact, actual run time might be much less than exponential in many cases, by some tweaking of the calculations.

4. PLANS

Some researches [9, 1] suggest policies that dictates different allowed states over time, step by step—*plans*. In our research, we extend the s-comb policies to allow definition of such plans, where each *stage* of the plan is defined by a single s-comb. We then suggest a continuous algorithm that

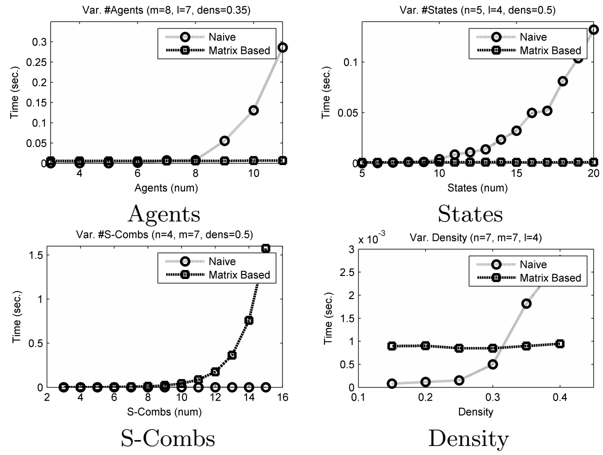

Figure 1: Runtime: Matrix-Based vs. Naïve

calculates the probability of no faults in the system, i.e., that all the agents are aligned to the same plan and stage. It hass time complexity of $O(mnd)$ and space complexity to $O(n(d+m))$, where d is the number of all the stages (in all plans) of the system.

5. REFERENCES

[1] B. Banerjee, L. Kraemer, and J. Lyle. Multi-agent plan recognition: Formalization and algorithms. In *AAAI'10*, pages –1–1, 2010.

[2] L. Comtet. *Advanced Combinatorics: The Art of Finite and Infinite Expansions*. Reidel, 1974.

[3] C. Dellarocas and M. Klein. An experimental evaluation of domain-independent fault-handling services in open multi-agent systems. In *Proceedings of the Fourth International Conference on Multiagent Systems (ICMAS-00)*, pages 95–102, 2000.

[4] B. Horling, V. R. Lesser, R. Vincent, A. Bazzan, and P. Xuan. Diagnosis as an integral part of multi-agent adaptability. Technical Report CMPSCI Technical Report 1999-03, University of Massachusetts/Amherst, January 1999.

[5] M. Kalech. Diagnosis of coordination failures: a matrix-based approach. *Autonomous Agents and Multi-Agent Systems*, 24(1):69–103, 2012.

[6] G. A. Kaminka and M. Tambe. Robust multi-agent teams via socially-attentive monitoring. *JAIR*, 12:105–147, 2000.

[7] M. Klein and C. Dellarocas. Exception handling in agent systems. In *Proceeding of the Third International Conference on Autonomous Agents*, pages 62–68, May 1999.

[8] M. Lindner, M. Kalech, and G. A. Kaminka. A representation for coordination fault detection in large-scale multi-agent systems. *AMAI*, 56(2):153–186, 2009.

[9] G. Sukthankar and K. Sycara. Hypothesis pruning and ranking for large plan recognition problems. In *Proceedings of the 23rd national conference on Artificial intelligence - Volume 2*, AAAI'08, pages 998–1003. AAAI Press, 2008.

[10] M. Tambe. Towards flexible teamwork. *Journal of Artificial Intelligence Research*, 7:83–124, 1997.