Issues with Methods for Scoring Competitors in RoboCup Rescue

James Parker, Julio Godoy, William Groves, and Maria Gini

Department of Computer Science and Engineering, University of Minnesota
`[jparker,godoy,groves,gini]@cs.umn.edu`

Abstract. As part of the RoboCup yearly competition, participants in the RoboCup Rescue Simulation League compete to solve a simulated urban search and rescue problem. For the last couple of years there has been debate about what metrics are best for selecting the competition winners. We statistically evaluate a variety of the metrics used and propose new ones,with the objective of determining which has the most power in determining the winner and the final rankings of the competitors. We also suggest a variety of convenient changes to the competition that would help increase the accuracy of the rankings.

1 Introduction

Robots are being deployed to solve a large variety of problems which are difficult for humans, such as wide-area surveillance, mapping dangerous caves or exploring the bottom of the ocean. Sometimes these robots need various degrees of autonomy and decision making if constant communication is infeasible, such as robots with limited battery power or the robot Curiosity on Mars. Our work focuses on the RoboCup Rescue Simulator, which simulates a disaster in an urban setting where first responders need to minimize the loss of infrastructure and lives.

The RoboCup Rescue Simulator is developed on a large scale, where programs must direct and coordinate up to 100 different first responders around the city. Coalition formation is known to be NP-complete [9], even when the tasks are known upfront. In the RoboCup Rescue Simulator, the locations of fires and civilians are not known until an agent gets close and perceives the event. This partial observability makes the problem much harder and can eliminate any hope of guaranteeing a bound on performance.

Evaluating the quality of solutions is very difficult in these types of domains, due to both the uncertainty and multiple goals. If more first responders search and rescue trapped civilians, then this will likely reduce the amount of infrastructure saved. A wide variety of different algorithms have been proposed and tested in the RoboCup Rescue Simulator, including distributed constraint optimization [10], auction based methods [1], partitioning of the city among the police agents [7], and evolutionary learning [6, 2]. This provides RoboCup Rescue with a diverse set of approaches to solve urban search and rescue which can all be directly compared and improved upon using the simulator.

The RoboCup Rescue Simulation competition is an event held at the large yearly RoboCup competition. In the RoboCup Rescue Simulation League simulations are run for a few days, and the best agents are announced. The ranking system for determining the winners has been questioned over the past few years because oddities caused unintuitive winners.

In this work, we explore what causes these oddities, compare old and new score aggregation methods, and propose new ideas on how to approach the problem to obtain a more statistically significant list of winners.

This paper is organized by first giving background on the RoboCup Rescue simulator and describing the goals of the competition. Next, we explore some issues using the RoboCup Rescue Simulation's metrics to make a direct assumption about the effectiveness of the solutions used by the competitors. A variety of methods used to evaluate algorithms in the competition are compared and the statistical power of each is evaluated. We end the paper with a review of the best methods investigated and some proposals to reduce the risk of misclassifying the best agents.

2 RoboCup Rescue Simulator

The RoboCup Rescue Simulator [5] was originally developed after an earthquake devastated Kobe, Japan. RoboCup Rescue tries to recreate the effects of earthquake on cities by trapping civilians under rubble, igniting fires across the city and blocking roads with rubble. Agents represented as the blue, red and white dots in Figure 1 are police, fire trucks and ambulances respectively. Police are able to clear the rubble off the roads to make them passable again. Fire trucks extinguish fires and prevent the fires from engulfing the city. Ambulances are able to dig out trapped civilians and bring them to a designated safe zone, the orange house icon in Figure 1. Buildings on fire increase in intensity as they turn yellow, orange and then red, and once they are completely burnt out and destroyed they turn black.

Communication is integrated into the RoboCup Rescue Simulator through both unlimited range transmission channels and local range shouting. The unlimited range transmission simulates radio channels, agents have to subscribe to a specific channel and then can only hear and send on that channel. The local voice communication does not require subscription, but both the long range channel and short range vocal have a simulated probability of transmission failure. There are two types of failures, either the whole message is lost or the message is sent to only a portion of people able to hear the message. Another limitation is that the long range channels have a fixed bandwidth, and if too many messages are sent some will be dropped.

As part of the yearly RoboCup competition, over a dozen teams develop competitive algorithms for agents to solve the urban search and rescue problems. These algorithms are tested against various map configurations over the course of a few days, split up into three elimination rounds: preliminaries, semi-finals

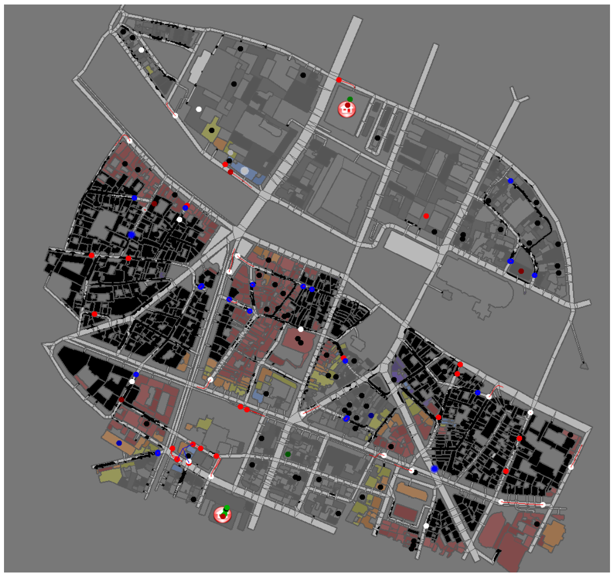


Fig. 1: The Paris RoboCup Rescue Simulation map.

and finals. At the end of each of these rounds, the lowest agents are dropped from the competition and all the scores are reset for the next round.

The score of each competitor at any time is computed as:

$$\sqrt{BD} \times (CA + CH), \tag{1}$$

where BD is the average percent of health of all buildings in the city, CA is the number of civilians alive and CH is the average percent health of civilians. During the simulation, there is no way to fix buildings or heal civilians, so the score is monotonically decreasing over time. Thus, the goal of the agents is to prevent the score from decreasing as much as possible. Police, who clear rubble to make roads passable, do not directly contribute to the score, yet are crucial for ambulances reaching damaged parts of the city and fire trucks having efficient paths to refill water. Due to the multiplication in the scoring function, it is important to balance both saving the city's infrastructure and rescuing civilians.

3 Scoring Issues

The score for a single competitor on a map is a good indication of performance, but when comparing multiple competitors some interesting issues can arise. Map configuration (and sometimes luck) can have a large impact on the score and the degree of score variation. This makes it harder to determine the real effectiveness of the algorithms of the different competitors without running a large number of games. In this section we explain cases when the score might not be a correct indicator of the efficiency of the algorithms used by a competitor.

Fires have a large impact in the RoboCup Rescue Simulator because they not only destroy buildings, but rapidly damage civilians and can be difficult to contain and extinguish. Although both civilian and buildings are accounted for in the score, the fires can have a detrimental unidirectional effect on saving civilians. In other words, if the fire trucks are inefficient and do very little to prevent the spread of fire, the ambulances will have a much harder time rescuing civilians because many will perish to fires before they can be reached. However, if ambulances are not rescuing civilians efficiently, the fire trucks' ability to extinguish fires is unchanged.

To compound the fire's effect on the score, completely extinguishing a fire becomes much more difficult as the number of burning buildings grows. This matter is made even more difficult by the partial observability in the scenario. A fire might grow a considerable amount before it is even noticed. Fires located in sections of the city which are surrounded by impassable roads can be especially difficult to deal with. These conditions can make the score very sensitive to a number of factors, such as the number of fire trucks, locations of fire, and road conditions.

An example of this is shown in Table 1, where the number of agents was adjusted on the same map (Mexico2) for four different competitors. Each program was run five times and the mean score computed. All competitors suffer greatly when going from 80 agents to 60 agents, but GUC_ArtSapience and Poseidon also suffer considerably more than S.O.S. when the agents are reduced from 100 to 80. These ranges provide the best comparison between different solutions' efficiencies, since agents can be fully utilized and have a large effect on the score.

Table 1: Scores of various competitors when agents are added or removed.

Total number of agents	GUC_ArtSapience	MRL	Poseidon	S.O.S.
160	268.31	271.85	269.66	272.01
140	266.67	270.49	258.18	272.11
120	256.94	251.52	245.29	269.00
100	221.63	230.83	207.05	255.66
80	160.79	182.72	140.87	211.74
60	60.63	55.73	50.21	80.88
40	48.05	47.93	47.01	47.21

Another issue highlighted by Table 1 is that at very high or very low number of agents, the map becomes too easy or too difficult respectively. When there are too many agents, there are not enough problems in the city to fully utilize all the agents, and this normalizes the scores between competitors. If the map configuration is too hard then all competitors score poorly causing the city to burn down for both the best and worst competitors.

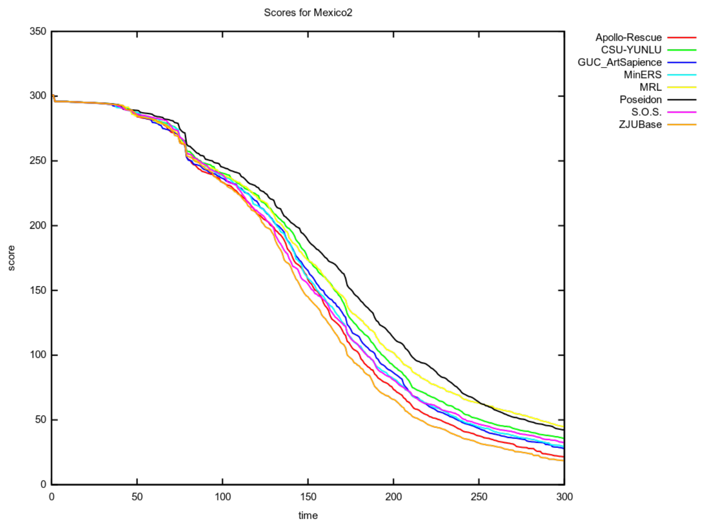
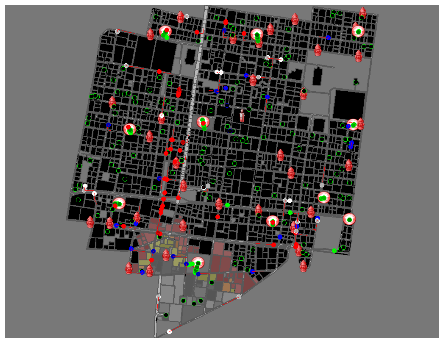
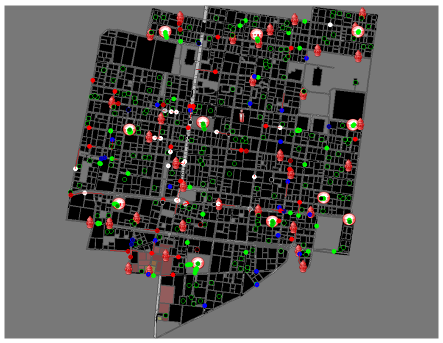


Fig. 2: Score time series of all agents run separately on a difficult Mexico City configuration.



(a) (a)



(b) (b)

Fig. 3: Final world states for the highest (a) and lowest (b) scoring competitors on a Mexico City map (Mexico2).

An example is highlighted on the Mexico City map in Figure 3 by showing many black burnt out buildings and expired civilians as black dots with green borders in both the best algorithm and the poorest. The scores of all eight competitors can be seen dropping together in Figure 2, causing all competitors to receive poor scores at the end. Since the RoboCup Rescue Simulation League is run as a competition, maps are created beforehand and without knowing the effectiveness of algorithms. Some maps with uninformative scores are inevitable, but these should not affect the overall results.

Another effect of fires is that their growth can create a bimodal distribution on the scores, separating competitors that were able to contain fires and those that were not. While there is no doubt that the competitors that contained the fires are better than those that did not, fire growth has such a compounding effect that the score differences might not be indicative of the efficiency differences. Figure 5 shows how the competitors are clearly separated into two groups and the fires cause a large difference in the scores. The worst competitor in the top group and the best competitor in the lower group are showed in Figure 5 (a) and (b), respectively. The difference in score between being able to control the fire in the center of the city and not is more than threefold between these agents, but this does not imply that the capability difference of these competitors is threefold in general. Fires grow exponentially [8] and are much more difficult to contain and extinguish the larger they get, but it is not effective to use all the fire trucks to extinguish a single fire because new fires, easy to extinguish, might start up unchecked and old fires also have a chance of reigniting. Therefore, agents must balance prioritization of large fires that require many agents to control with the discovery of small fires which can be rapidly extinguished by a single agent.

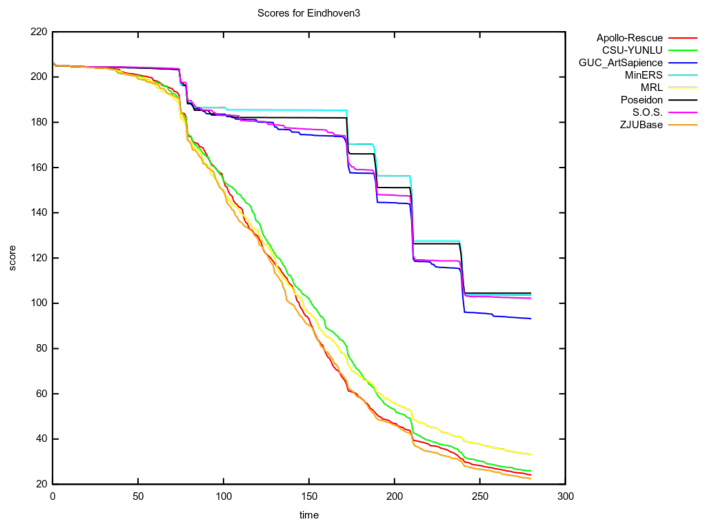
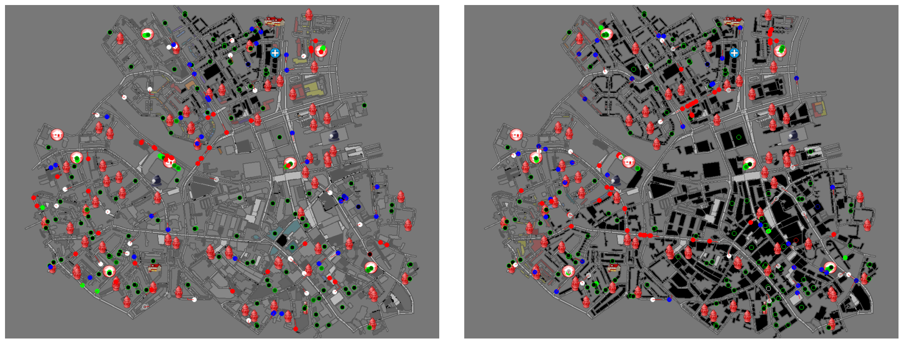


Fig. 4: A bimodal distribution of scores on a Eindhoven map.



(a) (a) (b) (b)

Fig. 5: The 4th and 5th best competitors, GUC_ArtSapience (a) and MRL (b), on a map of Eindhoven.

Many algorithms have probabilistic models built into them, so the variance of scores on the same map with the same configuration can be rather large. Table 2 and Table 3 show how on the "Paris4" map the standard deviation is practically half the estimated mean. This large variance is often due to the bimodal nature of whether the algorithm is able to contain the fires or not. In the competition, each map is only run once and as seen in "Paris4" in Table 2, the value generated at that single run can be substantially different than the algorithm's average performance.

4 RoboCup Rescue Point Assignments

Different maps have different numbers of civilians and buildings, which changes the range of scores available on that map. This makes it necessary to aggregate the scores across maps in a non-trivial way, since the value of the score across maps is not comparable. For example, if a competitor scored 10 on a map, it is unknown if this is a good score or not. If the map had only 10 civilians, then the score would start at 11 and would have only decreased to 10, suggesting that this competitor is quite good (or the map was easy). However, if the map had 300 civilians, then a score of 10 would be quite abysmal with almost all the city burnt down, almost all the trapped civilians perished or both.

A simple aggregate of the score would put more weight on maps with more civilians, which would give an unfair advantage to approaches that perform well in densely populated cities. The RoboCup Rescue Simulator committee has been aware of this fact, and placed ranks on algorithms for each map and then declared the winner the competitor with the highest summed rank. Recently, the competition committee has decided to move to a parametric model for determining winners under the assumption that more powerful conclusions can be drawn by utilizing the value of the scores. The term "score" refers to the performance of a single competitor on a single map as measured by Equation 1, while the

Table 2: Multiple trials of the Ri-one competitor.

Map	Kobe4	Berlin5	Eindhoven4	Eindhoven5	Istambul3	VC4	Paris4	Mexico3
Trial 1	38.769	31.334	4.064	39.212	0.391	5.955	29.917	34.591
Trial 2	58.979	26.629	3.597	40.333	0.37	7.191	58.001	31.848
Trial 3	48.794	28.231	3.984	38.511	0.369	4.885	11.013	31.841
Trial 4	57.26	26.676	3.918	39.862	0.366	5.702	22.336	31.849
Trial 5	35.148	27.558	3.98	39.878	0.366	5.337	25.808	31.836
Trial 6	46.616	25.326	3.962	39.987	0.368	5.512	22.132	31.854
Trial 7	54.38	25.661	3.862	39.988	0.366	6.001	56.443	31.847
Trial 8	59.524	27.818	3.765	42.067	0.366	4.749	29.61	31.841
Trial 9	54.554	23.795	3.875	38.082	0.37	4.983	20.026	31.849
Trial 10	42.805	33.831	3.962	39.324	0.367	4.384	24.775	31.841
Average	49.6829	27.6859	3.8969	39.7244	0.3699	5.4699	30.0061	32.1197
Stand. dev.	8.656	2.950	0.133	1.087	0.008	0.805	15.308	0.868
Competition score	47.350	36.500	4.138	39.155	0.391	6.385	9.614	33.021

term “points” refers to the value assigned to algorithms on maps to make scores across maps comparable. The rank of competitors is determined by summing up the points across all maps, so higher points correspond to a higher rank.

To analyze the effectiveness of different point metrics, we will compare the scores from the RoboCup Rescue Simulation League 2013 competition under the various metrics we present. We will evaluate the statistically significant information provided by each point metric and estimate the number of games needed to fully differentiate between all competitors. For the ranked point metric, the Friedman [4] test is performed in conjunction with the Wilcoxon signed-rank test [11] due to the nonparametric nature. The other point metrics are evaluated with the common ANOVA [3] and Student’s t-test. This will show the power of these point metrics and help decide which will be most beneficial to use.

Typically, the RoboCup Rescue Simulation League competition uses three rounds of elimination: a preliminary, semi-final and final round. At each round some of the agents are eliminated in a typical sport playoff. Scores are generated in a non-competitive manner because other competitors do not influence anyone else score, unlike most sport competitions where the opposing team has a significant effect on the score of a competitor. For this reason when evaluating statistical significance we will use data from all rounds of the competition instead of a single round to increase the number of data points.

In the 2013 RoboCup Rescue Simulation League competition, the top four competitors in descending order are:

Rank in the competition
GUC_ArtSapience; S.O.S.; MRL; Poseidon.

Table 3: Multiple trials of the MRL competitor.

Map	Kobe4	Berlin5	Eindhoven4	Eindhoven5	Istambul3	VC4	Paris4	Mexico3
Trial 1	89.412	15.057	3.327	35.686	0.34	4.371	6.246	31.847
Trial 2	68.112	14.542	3.327	34.316	0.34	4.259	6.249	31.847
Trial 3	80.046	15.296	3.417	34.308	0.34	4.37	6.531	31.847
Trial 4	92.253	14.357	3.417	34.314	0.34	4.482	18.289	31.847
Trial 5	82.096	14.98	3.327	35.015	0.34	4.371	8.379	31.847
Trial 6	91.234	15.007	3.327	35.575	0.34	4.482	7.26	31.847
Trial 7	75.927	14.422	3.327	34.334	0.34	4.483	16.749	31.847
Trial 8	77.909	16.082	3.327	35.517	0.34	4.259	9.118	31.847
Trial 9	91.747	14.357	3.327	34.36	0.34	4.371	7.135	31.847
Trial 10	90.723	14.806	3.417	35.153	0.34	4.482	7.631	31.847
Average	83.9459	14.8906	3.354	34.8578	0.34	4.393	9.3587	31.847
Stand. dev.	8.359	0.532	0.043	0.593	0.000	0.088	4.411	0.000
Competition score	93.882	14.16	2.861	37.216	0.361	5.529	10.085	32.058

GUC_ArtSapience's victory is primarily due to a low number of games and an outlier simulation. There were only six simulation trials for the final round, and on one of them GUC_ArtSapience received four times as many points as the second place competitor. The score of GUC_ArtSapience was indeed much higher in comparison to other scores, but if we removed this game the descending order of competitors is:

Rank removing one final game
S.O.S.; MRL and GUC_ArtSapience (tied for second); Poseidon.

If the points for all the rounds are included, as we do in our analysis, then the order will be:

Rank using points
S.O.S.; MRL; GUC_ArtSapience; Poseidon.

This is the order of competitors most commonly seen with the point systems.

4.1 Scoring using competitor ranking in each round

The RoboCup Rescue Simulation League used a rank sum to determine the winners until the 2012 competition. The strengths of using a ranked scheme is that very few assumptions need to be made out of the data. Specifically, when computing the statistical significance of results, there is no need to test the normality of residuals or the homoscedasticity.

However, using the ranked method does ignore the scale of scores within a map, which can lead to some unsettling results. For example, an algorithm can only slightly outperform a competitor on a little over half the maps, but lose horribly on all the other maps. With ranked aggregation this algorithm would perform better since it ranked higher, but from an intuitive perspective it seems that the competitor practically tied half the games and had significant winning margins on others and thus should win.

When evaluating all games, a rank sum aggregate rates competitors in descending order as:

Rank ranked sum
S.O.S.; MRL; Poseidon; GUC_ArtSapience.

Using the Friedman test, the χ^2 critical value found was 6.477, giving a p-value of 0.91. Such a high p-value indicates that the rankings are very likely to have been generated by chance. This means that even with 23 games between all rounds, it is very hard to tell if any of these algorithms actually performs differently. We went ahead and analyzed the pair-wise differences between algorithms using the Wilcoxon signed-rank test, despite having an elevated type I error. The p-values are displayed in Table 4. We can see that the S.O.S.-Poseidon pair of competitors has a statistically significant difference in scoring ability, and later tests confirm this pairing as significant, so it reduces the chance that this is a type I error. The S.O.S.-GUC_ArtSapience pair is also possibly different, since it is close to the 95% confidence level. However, for the other point methods the p-value is not nearly as close to 0.05 so it might be best to make no assumptions without further data. Assuming the data has a similar pattern of differences, it would take about 2500 simulations to reach a point where the competitors could all be differentiated with reasonable confidence.

Table 4: The p-value for pairwise comparison using ranking.

	S.O.S.	MRL	GUC_ArtSapience	Poseidon
S.O.S.		0.233	0.06	**0.025**
MRL			0.847	0.462
GUC_ArtSapience				0.694
Poseidon				

4.2 Normalized Points

As explained in Section 4, aggregating the scores directly is infeasible since the range of possible scores vary based on the map configuration. A simple transformation to make the scores comparable is to divide by the initial score, the maximum of each map, thus turning the score into a percent:

$$Points_i = \frac{Score_i}{InitialScore}.$$

This will reflect the percent of initial score the competitor managed to keep, which will have the side effect of giving more points to all agents on easier maps and fewer points to all competitors for harder configurations. Although these easy maps will make a larger difference in the point total, since all competitors will receive a similar amount on hard or easy maps what matters is the difference between competitors and not any points all of them receive together.

One issue is that the points will not reflect the capability difference in algorithms. For example on an easy map if algorithm A scores 0.95 and algorithm B scores 1.00 then the point difference will be 0.05 and indeed algorithm B is about 5% better. However, if algorithm A scores 0.05 and algorithm B scores 0.10, the difference is still 5% but algorithm B performed twice as well. We could remove this inflation on the points by removing the shared scores as such:

$$Points_i = \frac{Score_i - \min_j(Score_j)}{InitialScore},$$

where $\min_j(Score_j)$ is the score of the worst competitor. We will look at a similar transformation in Section 4.4 and discuss the differences there.

Using this normalized point method across all maps, the best estimate for the algorithms' performances in descending order is:

Rank using normalized points
S.O.S.; MRL; GUC_ArtSapience; Poseidon.

When using ANOVA to find our confidence in this ordering we must ensure the homoscedasticity of variance, which was tested with Levene's test. We found a p-value of 0.0054. This is well below the 5% confidence coefficient and means the variances are not equal and the results of ANOVA cannot be guaranteed. Ideally in this situation we would want to move to non-parametric tests, as we did in Section 4.1. Instead we will continue on and view our results of the paired Student's t-test shown in Table 5 skeptically, although the Central Limit Theorem does help strengthen these results due to the sample size of 23. We see that again there is a significant difference observed between the performance of S.O.S. and Poseidon. However, many of the other confidence levels are very low, especially between MRL and GUC_ArtSapience. If the scores continue in a similar pattern, we estimate it would take around 90000 simulations to confidently rank the ability of these four competitors. This is many more games required than the simple ranking method which uses less information, which shows the transformation from score to points must do more than simply make scores comparable.

4.3 Scaled Points

For the RoboCup Rescue 2013 competition, the committee decided to use a scaling approach off the best competitor to normalize scores between maps. In this approach, the best algorithm receives a fixed number of points and all other competitors receive points so that the ratio between the best competitor's score and theirs is approximately proportional to their ratio of points. A simple

Table 5: The p-value for pairwise comparison using normalized points.

	S.O.S.	MRL	GUC_ArtSapience	Poseidon
S.O.S.		0.238	0.131	**0.031**
MRL			0.966	0.394
GUC_ArtSapience				0.322
Poseidon				

example would be if algorithm A scored 10 and 20 on two maps respectively and algorithm B scored 15 and 15. Suppose the fixed point amount for the winning algorithm is 100, then the points assigned to algorithm A would be 67 and 100, while algorithm B would receive 100 and 75 points respectively. Formally, the points for algorithm i can be computed as the following formula on a map:

$$Points_i = MaxPoints \times \frac{Score_i}{\max_j(Score_j)}, \tag{2}$$

where $MaxPoints$ is the fixed amount of points for the winning algorithm and $\max_j(Score_j)$ is the score of the winning algorithm.

In addition to the issues with the 2013 point method outlined in Section 4, points were forced to be a whole number without ties and the fixed point amount was too small. The whole number round from the example in the previous paragraph does not look bad since algorithm A receives 67 instead of simply 66.666. When the fixed number is much lower and there are more algorithms, the discretization and lack of allowing ties can cause the points to become skewed. If algorithm A, B, C and D score 100, 98, 96 and 70 respectively with only a fixed winning amount of 10, then the algorithms would be awarded 10, 9, 8 and 7 points respectively. Both algorithms B and C are not awarded as many points as they should have, and from the point breakdown it seems algorithms D and C scored much closely than they actually did.

This is a very preventable problem by simply not restricting points to whole numbers or making the fixed winning amount a large whole number. For our experiments, we simply used the scaled point method outlined in Equation 2 and allowed real numbers to reduce errors from discretization. The best competitors in descending order using this scoring are exactly the same as the normalized point system from Section 4.2:

Rank using scaled points
S.O.S.; MRL; GUC_ArtSapience; Poseidon.

In the 2013 competition considering just the final six simulations gave the ordering:

Rank using final games only
GUC_ArtSapience; S.O.S.; MRL; , Poseidon.

If real numbers are used instead of the non-tied whole numbers and also considering just the final games, the best ordering of competitors is:

Rank using real numbers
S.O.S.; GUC_ArtSapience; MRL; Poseidon.

While most orderings place GUC_ArtSapience below MRL, this does rank S.O.S. as first where it is positioned in all orderings using the full 23 simulations.

The Levene's test also had a low p-value of 0.0294, so we are unable to trust the power from ANOVA. Again, in this case we should perform the tests from Section 4.1 instead of continuing with parametric tests, but we will for speculation. Table 6 shows that again there is a significant difference between S.O.S and Poseidon without being able to draw any other definitive conclusions.

Table 6: The p-value for pairwise comparison using scaled points.

	S.O.S.	MRL	GUC_ArtSapience	Poseidon
S.O.S.		0.328	0.248	**0.029**
MRL			0.945	0.437
GUC_ArtSapience				0.447
Poseidon				

If the scores followed a similar pattern, it would take approximately 40000 simulations to reach a high level of confidence in our ranking of competitors. While this is still a very high number, it is less than half of the 90000 simulations required for the normalized point system from Section 4.2. This large reduction can be attributed to the normalization between easy games and hard games, because the points are not scaled on the maximum possible points but instead the maximum achieved points by the competitors. This time when algorithm A gets 95 and 5 points on two maps and algorithm B gets 100 and 10 respectively, algorithm B will get many more points on the second map due to the large relative difference between scores.

4.4 Normalized Scale

This year, the RoboCup Rescue Simulator committee decided to score the 2014 competition with the following method:

$$Points_i = \frac{Score_i - \min_j(Score_j)}{\max_k(Score_k) - \min_j(Score_j)}.$$

This is a mix of both the normalized and scaled method from Sections 4.2 and 4.3 because it forces the range of points to be between 0 and 1 and guarantees that both 0 and 1 will be scored. There is one exception when all agents have exactly the same score, then the denominator is 0 and this game will simply be dropped from consideration. In this point system the best algorithm is always given 1 point and the worst algorithm is given 0, with the remaining algorithms making a line between these two.

The Levene's test this time gives $p = 0.069$ and we assume that the variances are similar enough to proceed with parametric tests. ANOVA gives a p-value of 0.488 which means it is unable to say that one algorithm is different from all others. The Student's t-test shown in Table 7 for all pairs is weaker than all the pairwise comparisons, because the S.O.S.-Poseidon pair is not statistically different this time. This might imply that the previous parametric comparisons yielded a type I error as the failure of Levene's test did not guarantee the results. The Wilcoxon signed-rank test also had an elevated type I error due to running the test on a family of pairs, but since the S.O.S.-Poseidon pair has always had the lowest p-value this is still the most different results we can find. If we project that a similar scoring pattern (will be used), then this point system will take around 1500 simulations to confidently order the algorithms. This is the fewest expected simulations needed, with the ranked scheme as the next closest at around 2500 simulations. Similarly to all other parametric point systems, the ordering for best algorithms is:

Rank in competition
S.O.S.; MRL; GUC_ArtSapience; Poseidon.

Table 7: The p-value for pairwise comparison using both normalized and scaled points.

	S.O.S.	MRL	GUC_ArtSapience	Poseidon
S.O.S.		0.608	0.352	0.107
MRL			0.741	0.363
GUC_ArtSapience				0.464
Poseidon				

5 Conclusions

From the point systems investigated, the ranking and the proposed 2014 competition from Sections 4.1 and 4.4 are most appealing. Although it is doubtful that enough simulations can be run to reach statistical significance, both these approaches require significantly fewer trials and will provide more confidence in the estimated algorithm order. Unfortunately these two methods estimate different orderings of algorithms, although none of the algorithms have strong evidence of being different than the others. Our investigation also shows that discretization should be avoided by using either large whole numbers or real numbers when representing points in parametric approaches.

One easy way to increase the number of simulations would be to utilize the data from previous rounds as we did, because the algorithms do not interfere with each other's scores. While this RoboCup Rescue Simulation League is part of RoboCup, which utilizes sport elimination techniques to help create excitement and interest in the competitions, using all the finalist's data to determine the top four rankings will not interfere with the round scheduling or elimination process. Some competitors change their algorithms between the rounds, and from a statistical point of view this will cause technical issues, but unless there is a devastating bug their score distribution probably will not change significantly. Competitors who do have devastating bugs normally get eliminated so aggregating earlier should not penalize competitors any more than the system already in place.

Another way to increase games without drastically changing the structure of the competition would be to run each algorithm on each map multiple times instead of just once. This would help reduce the variation of algorithms within a single map to give a more accurate estimation of its true effectiveness. Active simulations are shown on monitors, but the extra simulations could be run on clusters not connected to the monitor. Although it might be unrealistic to reach statistical significance during the few competition days, it would be helpful to aggregate enough games so that the ordering of competitors does not change when any single game is excluded from the counted set.

Analysis of statistical significance can still be used even if the exact ordering of all algorithms is unclear. For the elimination rounds instead of cutting in a semi-arbitrary way the amount of competitors each round, only agents that are significantly behind could be dropped. If the top four estimated competitors are put into an equivalence class, then if any of the other competitors is pairwise statistically worst than all four of these competitors they will be eliminated at the end of the day. This could very easily work for the preliminary round, but there might be more than four agents left in the final round that are not statistically different.

References

1. A. Bredenfeld et al. *RoboCup 2005, LNAI 4020.* Springer, 2006.
2. F. dos Santos and A. L. C. Bazzan. Towards efficient multiagent task allocation in the RoboCup Rescue: a biologically-inspired approach. *Journal of Autonomous Agents and Multi-Agent Systems*, 22(3):465–486, 2011.
3. R. A. Fisher. The correlation between relatives on the supposition of Mendelian inheritance. *Transactions of the Royal Society Edinburgh*, 52:399–433, 1918.
4. M. Friedman. The Use of Ranks to Avoid the Assumption of Normality Implicit in the Analysis of Variance. *Journal of the American Statistical Association*, 32(200):675–701, 1937.
5. H. Kitano and S. Tadokoro. RoboCup rescue: A grand challenge for multiagent and intelligent systems. *AI Magazine*, 22(1):39–52, 2001.
6. I. C. Martínez, D. Ojeda, and E. A. Zamora. Ambulance decision support using evolutionary reinforcement learning in RoboCup rescue simulation league. In *RoboCup 2006: Robot Soccer World Cup X*, pages 556–563. Springer-Verlag, 2007.
7. M. Nanjanath, A. Erlandson, S. Andrist, A. Ragipindi, A. Mohammed, A. Sharma, and M. Gini. Decision and coordination strategies for RoboCup rescue agents. In *Proc. SIMPAR*, pages 473–484, 2010.

8. J. Parker and M. Gini. Teams to exploit spatial locality among agents. In *Spatial Computing Workshop at AAMAS*, 2013.
9. T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohmé. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1–2):209–238, 1999.
10. P. Scerri, A. Farinelli, S. Okamoto, and M. Tambe. Allocating tasks in extreme teams. In *Proc. Int'l Conf. on Autonomous Agents and Multi-Agent Systems*, pages 727–734, 2005.
11. F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, Dec. 1945.