Run-Time Analysis of Classical Path-Planning Algorithms

Pablo Muñoz, David F. Barrero and María D. R-Moreno

Abstract Run-time analysis is a type of empirical tool that studies the time consumed by running an algorithm. This type of analysis has been successfully used in some Artificial Intelligence (AI) fields, in paticular in Metaheuristics. This paper is an attempt to bring this tool to the path-planning community. To this end the paper reports a run-time analysis of some AI classical algorithms applied to solve the path-planning problem. In particular, we analyse the statistical properties of the run-time of the A*, Theta* and S-Theta* algorithms with a variety of problems of different degrees of complexity. We conclude that the time required by these three algorithms follows a lognormal distribution. In low complexity problems, the lognormal distribution looses some accuracy to describe the algorithm run-times. The lognormality of the run-times opens the use of powerful parametric statistics to compare execution times, which could lead to stronger empirical methods.

1 Introduction

Path-planning is a well known problem in Artificial Intelligence (AI) with many practical applications. Several classical AI algorithms, such as A* [12], have been applied to solve the path-planning problem as the notable literature about this topic shows. Nonetheless, despite the research interest that this issue has attracted, the statistical properties of the run-time of these algorithms has not been, to the authors' knowledge, studied before.

Pablo Muñoz
Departamento de Automática, Universidad de Alcalá, Spain, e-mail: pmunoz@aut.uah.es

David F. Barrero
Departamento de Atuomática, Universidad de Alcalá, Spain e-mail: david@aut.uah.es

María D. R-Moreno
Departamento de Automática, Universidad de Alcalá, spain, e-mail: mdolores@aut.uah.es

The study of the run-time from a statistical perspective is known as *run-time analysis*. The common practice in the literature is to report the run-time of the algorithm with means and, sometimes, some dispersion measure. However, this practice has several drawbacks, mainly due to the loose of valuable information that this reporting practice involves such as asymmetries in the run-time, or the shape of its distribution.

Run-time analysis overcomes this limitation by considering all the data with its focus on the statistical properties of the algorithm run-time. The main tool of run-time analysis is the *Run-Time Distribution* (RTD) which is the empirical statistical distribution of the run-time; in this way, the RTD is fully characterized. Other statistics such as the mean can be obtained from the RTD, and it opens exciting parametric statistical analysis tools. For instance, once the run-time distribution of an algorithm is identified, parametric hypothesis tests can be used to compare the run-time of two algorithms, providing a more rigorous comparison methodology.

This paper is an attempt to provide a first run-time analysis of some classical path-planning algorithms (A* [12] and Theta* [18]), and a new path-planning algorithm (S-Theta*) that we have developed. We follow an empirical approach, running the algorithms in grid-maps of different difficulties and fitting the resulting run-time distribution with some statistical distributions. We show that, in the same line as the related run-time literature in other fields, the run-time of the three algorithms under study follow a lognormal distribution. We also observe a dependence between the run-time distribution and the problem difficulty. Our experiments show that very easy problems are not well characterized by a lognormal RTD.

The paper is structured as follows. First, we introduce the path-planning algorithms that are object of study. Then we describe the RTD analysis, including some related literature mainly from Metaheuristics. Next, in section 4 we perform the run-time analysis. The paper finishes with some conclusions.

2 Path planning

Path-planning or pathfinding [19] is a widely discussed problem in robotics and video games [11, 17]. In a simple manner, the path-planning problem aims to obtain a feasible route between two points. Feasible means that the route can not violate constraints such as traversing obstacles. This is a classical problem that can be addressed with several search algorithms such as classical graph-search [19], Evolutionary Algorithms [20] or multiagent systems [8], just to mention some of them.

In this work we have considered one classical algorithm, A* [12], and two variations. These algorithms use a grid of cells to discretize the terrain. And of course, they work over nodes. A* is a simple and fast search algorithm that can be used to solve many AI problems, path-planning among them. It combines an heuristic and a cost function to achieve optimal paths. However, it has an important limitation: it typically uses 8 neighbours nodes, so it restricts the path headings to multiples of $\pi/4$, causing that A* generates a sub-optimal path with zig-zag patterns. Other ap-

proximations use more adjacent nodes or use framed cells [1] to solve (or relax) this limitation, and thus requiring, in most cases, more computational effort. For each node, A* maintains three values: (i) the length of the shortest path from the start node to actual node; (ii) the heuristic value for the node, an estimation of the distance from the actual node to the goal; and (iii) the parent of the node. The heading changes limitation makes that the best heuristic for A* is the octile distance.

There are some variations of A* that try to generate smoother heading changes. Some proposals smooth the path using framed cells [1] while others post-process the result of A* [4]. Another group of algorithms called "any-angle" allow heading changes in any point of the environment. Most popular examples of this category are Field D* [9]. Theta* [18] allows any node to be the parent of a node, not only its predecessor like A*. An alternative to Theta* is our algorithm called S-Theta*, which modifies the cost and heuristic functions to guide the search in order to obtain similar paths to Theta*, but with less heading changes. An exhaustive comparison over some path-planning algorithms can be found in [6].

Given its popularity and presence in the literature, we selected A*. It is a very well studied algorithm and it is quite popular in path-planning research, so it seemed reasonable to chose it. Additionally we have also included Theta* and S-Theta*, which are two algorithms designed specifically to path-planning in order to solve the heading changes problem. Before begining the RTD analysis of those algorithms, we first introduce in more detail the RTD analysis and related literature.

3 Introduction to run-time analysis

The basic tool used for run-time analysis is the RTD, term that was introduced by Hoos and Stützle [16]. Let us name $rt(i)$ the run-time of the ith successful run, and n the number of runs executed in the experiment, then the RTD is defined as the empirical cumulative probability $\hat{P}(rt \leq t) = \#\{i|rt(i) \leq t\}/n$ [15]. It is simply the empirical probability of finishing the algorithm run before t. There is an extensive literature about RTDs, mainly in Metaheuristics, but there are also several studies in classical AI problems.

Hoos and Stützle applied RTD analysis to different algorithms and problems. They found that the RTD of WSAT algorithms applied to the 3SAT problem is exponential when the parameters setting is optimal, shifted exponential or Weibull when the parameters setting is not optimal [16]. Analogously, they studied in [13] the RTD of some other stochastic local search algorithms, such as GWSAT, GSAT with tabu-lists, TMCH and WMCH, to solve instances of SAT and CSPs, finding that, again, when parameters are optimal, the RTD follows an exponential, and otherwise RTD fits a Weibull distribution. Curiously, this result only holds for hard instances, in easy instances they did not find statistical significance. In a later work [14], they observed that the RTD of hard 3SAT instances solved with WalkSAT algorithm also follows an exponential distribution, and more interestingly, the higher the difficulty of the problem, the higher the fit is found. In a more recent work, Hoos and

Stützle [15] compared various versions of GSAT and WalkSAT algorithms to solve some problems coded as 3SAT (random 3-SAT, graph coloring, block world and logistic planning), finding that these algorithms also have exponential RTDs for hard problems and high values of noise parameter. More importantly, they found that RTDs of easy problems are no exponential, despite their tails are still exponential.

Chiarandini and Stützle [5] studied the RTD of ILS, ACO, Random Restart Local Search and two variants of SA applied to the course timetabling problem, finding that the Weibull distribution approximates well the RTDs. They report, however, that in SA, the RTD in hard problems can be approximated, at least partially, using a shifted exponential distribution. On the contrary than Hoos, Stützle and Chiarandini, Frost *et al*. [10] studied the RTD using the same algorithm, backtracking, with different problem instances of the CSP. The RTD of the algorithm running on solvable instances [7] follows a Weibull distribution, while unsolvable instances generate lognormal run-times. However, only the lognormal distribution for solvable problems had statistical significance. In addition, Barrero *et al*. studied the RTDs in tree-based Genetic Programming [3], finding lognormal RTDs whose goodness of fit depends on the problem difficulty.

In summary, we conclude that despite the variety of algorithms and problems studied using RTDs, the are three omnipresent distributions: Exponential, Weibull and Lognormal. Curiously, these three distributions play a central role in Reliability Theory, suggesting a link. In the following section we study the presence of these three distributions in the RTD of A*, Theta* and S-Theta* for path-planning problems of varying difficulty.

4 RTD analysis of path-planning algorithms

In order to carry out the run-time analysis, we need to gather empirical data in a controlled way. Given the need of a uncertainty source, experiments may vary two factors, the random seed and the problem. The former is common in the Metaheuristics and random search literature, while the latter is used with deterministic algorithms. Even though this distinction does not use to be clear in the literature, we think that it is critical since it determines the comparability of the experiments and their interpretation.

4.1 Experimental design

In order to assess the run-time behavior of the path-planning algorithms we have performed an experimental approximation. These algorithms are deterministic, so, on the contrary than other run-time analysis performed in Metaheuristics, there is no variability due to the random seed. In this case, the variation comes from the problem, to be more specific, we used a random map generator that can control the

percentage of obstacles in the map, and therefore, we can set the complexity of the problem. It provides a mechanism to study how the problem difficulty influences the run-time behavior of the algorithms[1].

We have generated random maps with different ratio of random obstacles, in particular we used $5\%, 10\%, 20\%, 30\%$ and 40% of obstacles. For each ratio of obstacles we generated $2,000$ random maps of 500×500 nodes and solved the map with each of the three algorithms under study. This procedure yields $2,000 \times 5 = 10,000$ runs for each one of the three algorithms. The initial point in the map is always the corner at the top left of the map, and the goal node is placed at the right, locating between the bottom corner and randomly 20% up nodes. The map generator was implemented with guarantees to keep at least one path from the start node to the goal node. To do this, when an obstacle is set, his periphery is protected to avoid overlapping obstacles. Thus, it is always possible surround an obstacle to avoid it.

The algorithms were implemented in Java. In order to make a fair comparison, the implementation of the three algorithms use the same methods and structures to manage the information grid[2]. To measure the runtime we have employed `System.currentTimeMillis()`. Reporting time in this way has several drawbacks [2], but in this case we think it is justified because the algorithms contains computations that are not well captured by machine-independent time measures, such us the number of expanded nodes. Of course, the price we have to pay is an increased difficulty to repeat and compare these results, but given that our interest is studying the statistical properties of the run-time rather than compare algorithms, we think that in this case it is an acceptable drawback.

4.2 Experimental results

We have firstly performed an exploratory analysis of the results. To this end we depicted several histograms of the run-times, as can be seen in Fig. 1. The histograms were grouped by algorithm and problem hardness and they show some interesting facts.

We observe that the shape of the run-time histograms varies in function of the algorithm and problem hardness. S-Theta* has a longer tail and its shape is more asymmetrical in comparison to the rest of the algorithms, this fact is more clear in hard problems than in easy ones. The run-time of A* presents a smaller variance than the other two algorithms, and it increases with the problem hardness. The run-time required to solve hard problems has more variance than easy problems. In any case, the shape and range of values of Theta* and S-Theta* are similar, which seems reasonable given that they are variations of the same algorithm.

---

[1] In case of acceptance, all the code, datasets and scripts needed to repeat the experiments reported in this paper would be published on a web site under an open licence.

[2] The execution was done on a 2 GHz Intel Core i7 with 4 GB of RAM under Ubuntu 10.10 (64 bits)

Looking for a statistical distribution able to fit data is more interesting. With the results reported in the literature we have tried to fit data to the three distributions that appear to play a more important role in RTD analysis, the Exponential, Weibull and Lognormal distribution. So, the histograms shown in Fig. 1 are depicted with a set of overlapping distributions, which corresponds to the three statistical distributions previously mentioned. The parameters of these distributions were fitted by maximum likelihood. As the reader can observe in Fig. 1, the distribution that fits better our data in most cases is the lognormal. The exceptions to this observation are Theta* and S-Theta* solving maps with a very low number of obstacles; in that case the exponential distribution seems to describe the run-time behavior better than the

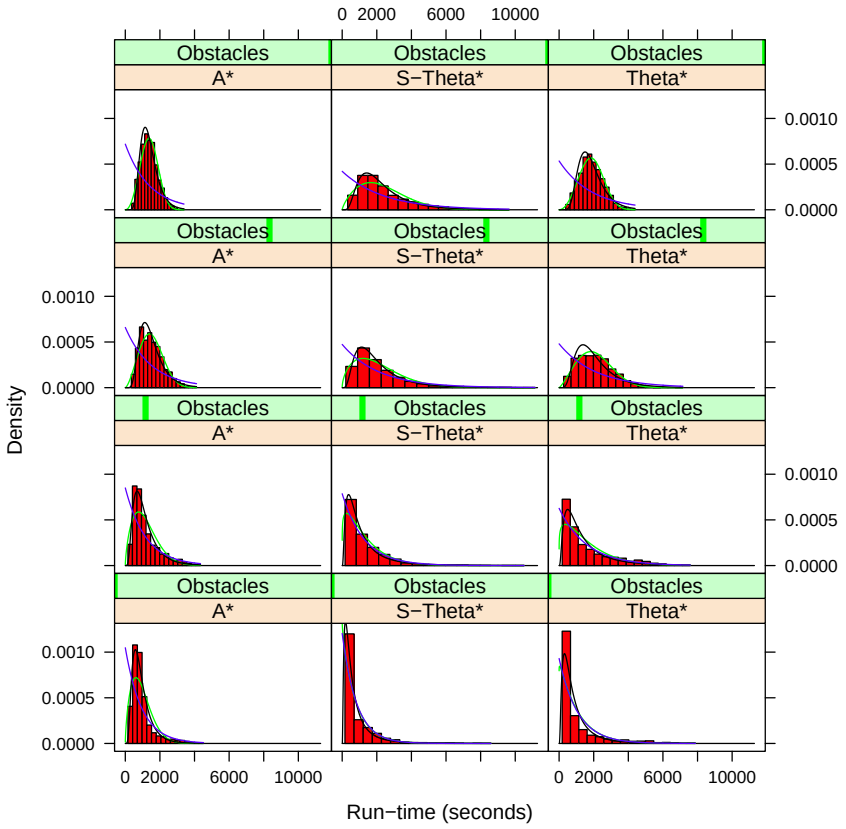

Fig. 1 Histogram of the run-time, measured in seconds, of the three path-planning algorithms under study. Histograms have been grouped by algorithm and ratio of obstacles. The histograms have been adducted with three distributions that appear overlapped: Lognormal (black), exponential (blue) and Weibull (green)

lognormal. Even in this case, the A* run-time is well described by the lognormal, that is able to describe its peaks better than the exponential.

Table 1 Estimated parameters of the lognormal distribution fitted by maximum likelihood. Each row corresponds to an experiment with 2,000 problems with the given rate of obstacles in the map

Algorithm	Obstacles	$\hat{\mu}$	$\hat{\sigma}$
A*	5	6.690	0.567
A*	10	6.886	0.600
A*	20	7.131	0.542
A*	30	7.228	0.449
A*	40	7.176	0.361
Theta*	5	6.520	0.883
Theta*	10	6.989	0.887
Theta*	20	7.395	0.700
Theta*	30	7.515	0.534
Theta*	40	7.464	0.390
S-Theta*	5	6.202	0.975
S-Theta*	10	6.749	0.916
S-Theta*	20	7.220	0.744
S-Theta*	30	7.445	0.648
S-Theta*	40	7.601	0.589

Given the observations of the exploratory analysis, it seems reasonable to focus the study on the lognormal distribution and hypothetise that the run-time of the A*, Theta* and S-Theta* algorithms in path-planning problems follow a lognormal distribution. The parameters of the lognormal distributions that we obtained using maximum-likelihood are shown in Table 1. In order to evaluate the hypothesis about the RTDs lognormality, it is desirable to provide additional evidences. One of the most interesting properties of the lognormal distribution is its close relationship to the normal distribution, actually, lognormal data can be converted to normal data by simply taking logarithms. We can use this property to verify whether the RTD is lognormal or not in a more formal way.

To verify the lognormality of the RTD, we first plotted a QQ-plot of the logarithm of the run-time against a normal distribution, which is shown in Fig 2. If the logarithm of the run-time is normal, we can conclude that the run-time is lognormal. Fig. 2 confirms our initial suspicion about the lognormality of the RTD. In addition, the QQ-plots also show the relationship between the distribution and the problem hardness. Theta* and S-Theta* algorithms produce less lognormal RTDs in very easy problems. On the contrary, the influence of the problem hardness on A*, if it has any, is not so evident, the QQ-plot is almost a line for all the ratios of obstacles. In summary, A* RTDs seem lognormal for any number of obstacles, while the run-time of Theta* and S-Theta* algorithms seem to follow a lognormal distribution, but easy problems fit worse.

A more rigorous analysis of the lognormality of the run-time is desirable. For this reason we have performed a Shapiro-Wilk test of normality of the logarithm of

the run-time, whose result is shown in Table 2. In order to avoid undesirable effects of the large number of samples, we have computed the test using 30 random runs. Results shown in the table confirms our previous observations in the histograms and the QQ-plot. The p-values in almost all the cases are quite high, which means that the hypothesis of lognormality is compatible, with a high probability, with our data. However, there are two exceptions, the p-value of Theta* and S-Theta* with a ratio of 5% of obstacles is quite small, 0.0083 and 0.0003 respectively, which drives us to reject the null hypothesis (i.e. the lognormality of the data) with $\alpha = 0.001$.

We were curious about the different RTDs reported by the literature and the reason of those differences. In order to obtain some clue for its answer, we performed a simple experiment. Instead of plotting run-time histograms grouped by algorithm and problem hardness, we tried to plot histograms joining all the runs belonging to

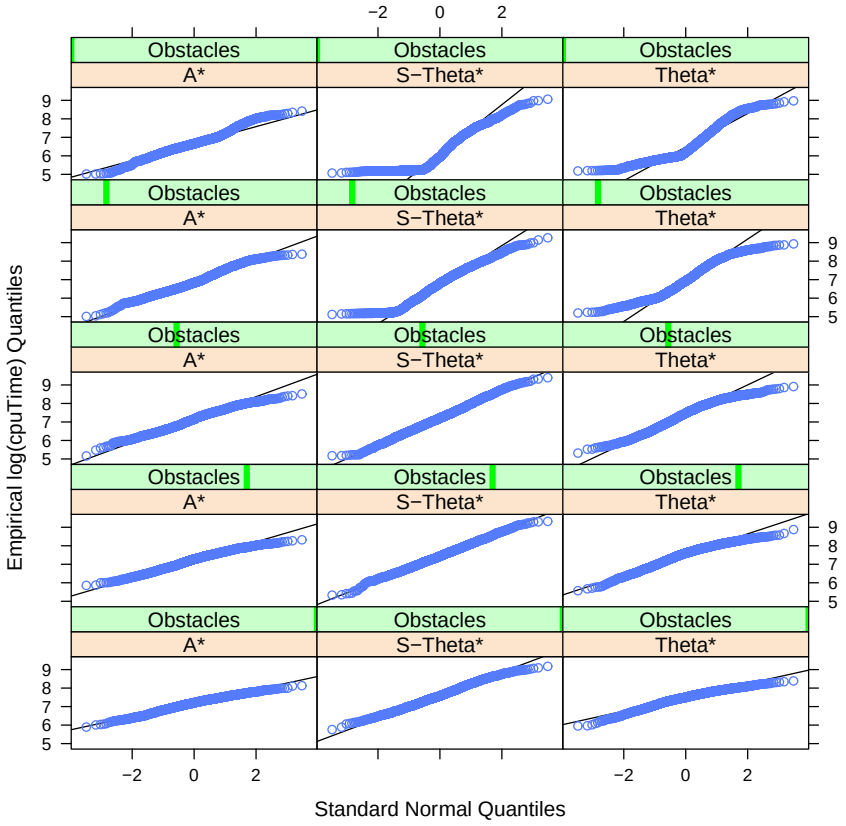

Fig. 2 Quantile plot that assesses the normality of the logarithm of the run-time of the three algorithms under study: A*, Theta* and S-Theta*. Given the relationship between the lognormal and normal distributions, the logarithm of lognormal data must transform it into normal data

Table 2 Shapiro-Wilk test of normality of the logarithm of the run-time. With these p-values we cannot reject the hypothesis of normality of the logarithm of the run-time, which means that we cannot reject the lognormality of the run-time. Only Theta* and S-Theta* with a ratio of 5% of obstacles provide evidence to let us reject the normality of the RTD

Algorithm	Obstacles	W	p-value	Significance
A*	5	0.9772	0.7473	
A*	10	0.9579	0.2743	
A*	20	0.9808	0.8475	
A*	30	0.9378	0.0792	
A*	40	0.9546	0.2237	
Theta*	5	0.8998	0.0083	$\alpha = 0.001$
Theta*	10	0.9437	0.1142	
Theta*	20	0.9479	0.1487	
Theta*	30	0.9343	0.0641	
Theta*	40	0.9444	0.1194	
S-Theta*	5	0.8322	0.0003	$\alpha = 0.001$
S-Theta*	10	0.9409	0.0965	
S-Theta*	20	0.9771	0.7428	
S-Theta*	30	0.9829	0.8968	
S-Theta*	40	0.9876	0.9725	

the same algorithms, in this way, problems of different hardness were merged. As in the exploratory analysis, we plotted the three main distributions in RTD literature to check rapidly whether data fits or not that distribution. The result can be seen in Fig. 3. We have to consider that this figure shows in fact an overlapping of several distributions, it can be seen as the sum of each column in Fig. 3.

Fig. 3 is a quite interesting result. The lognormal distribution still seems to fit very well the A* run-time, however, the Theta* algorithms are not so clear. The left tails of their histogram seems to have disappeared, or at least it is sufficiently small to not appear clearly in the histogram. This fact introduces the exponential distribution in the discussion, it can be seen that this distribution is able to fit data quite well, nonetheless, the lognormal distribution still provides a excellent fit. So, it makes us conjecture that depending on how the run-time is visualized the statistical model that fit the RTD may change. In particular, it is well known that joining random variables of a certain distribution may produce a random variable of another distribution. This fact could explain, in part, the diversity of RTDs found in the literature, and it is an additional motivation to take care about how experimentation and data processing are done.

With this evidence, it seems reasonable to assume the lognormality of the RTD in problems with a ratio of obstacles higher than 5% of the three algorithms under study.

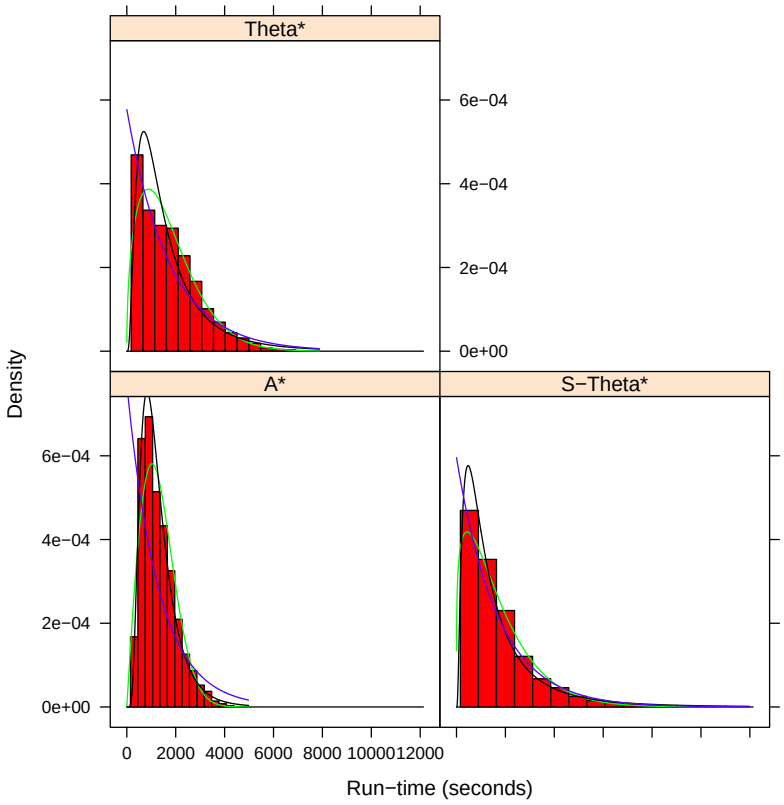

Fig. 3 Histogram of the run-time, measured in seconds, of the three path-planning algorithms under study. Histograms have been grouped by algorithm, but not by ratio of obstacles, so each histogram contains runs of different problem hardness. The histograms have been adjusted with three distributions that appear overlapped: Lognormal (black), Weibull (green) and exponential (blue)

5 Conclusions

Along this paper we have performed a run-time analysis of A*, Theta* and S-Theta* to study their RTD statistical properties and the influence of the problem hardness. The RTD of those algorithms, when applied to a set of non-trivial path-planning problems of similar hardness, follows a lognormal distribution. The evidence we have found in this line is strong. However, we also observed that the goodness-of-fit is weaker for the Theta* algorithms when the problem is easy.

This observation leads to a better knowledge about A*, Theta* and S-Theta algorithms, but it also has a practical implications. The common procedure to compare run-times followed in the literature is a naïve comparison of means, which is a

weak method from a statistical point of view. If the RTD of the algorithm is known, strong parametric methods could be used, and in particular lognormal RTDs open the use of well-known (and easy) statistics to enhance the experimetal methods used to study path-planning algorithms. So, in order to compare the run-time of two algorithms with a sound statistical basis we can -and should- use hypothesis testing (Student's t-test with the logarithm of the run-time) or ANOVA if several algorithms are involved.

From a more general perspective, our results are clearly aligned with previous results reported in the literature. Very different algorithms applied to different problems have shown a similar run-time behaviour, which turns out an intriguing fact. So, a natural question that raises at this point is whether this behaviour is as general as it seems to be and, more importantly, why RTDs are so well described by only three distributions. These are questions that we think deserve some research.

Acknowledgements Pablo Muñoz is supported by the European Space Agency (ESA) under the Networking and Partnering Initiative (NPI) *Cooperative systems for autonomous exploration missions*.

References

1. G. Ayorkor, A. Stentz, and M. B. Dias. Continuous-field path planning with constrained path-dependent state variables. In *ICRA 2008 Workshop on Path Planning on Costmaps*, May 2008.
2. R. Barr and B. Hickman. Reporting Computational Experiments with Parallel Algorithms: Issues, Measures, and Experts' Opinions. *ORSA Journal on Computing*, 5:2–2, 1993.
3. D. F. Barrero, B. Castaño, M. D. R-Moreno, and D. Camacho. Statistical Distribution of Generation-to-Success in GP: Application to Model Accumulated Success Probability. In *Proceedings of the 14th European Conference on Genetic Programming, (EuroGP 2011)*, volume 6621 of *LNCS*, pages 155–166, Turin, Italy, 27-29 Apr. 2011. Springer Verlag.
4. A. Botea, M. Muller, and J. Schaeffer. Near optimal hierarchical path-finding. *Journal of Game Development*, 1:1–22, 2004.
5. M. Chiarandini and T. Stützle. Experimental Evaluation of Course Timetabling Algorithms. Technical Report AIDA-02-05, Intellectics Group, Computer Science Department, Darmstadt University of Technology, Darmstadt, Germany, April 2002.
6. K. Daniel, A. Nash, S. Koenig, and A. Felner. Theta*: Any-angle path planning on grids. *Journal of Artificial Intelligence Research*, 39:533–579, 2010.
7. S. Epstein and X. Yun. From Unsolvable to Solvable: An Exploration of Simple Changes. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*, 2010.
8. M. Erdmann and T. Lozano-Perez. On multiple moving objects. *Algorithmica*, 2:477?–521, 1987.
9. D. Ferguson and A. Stentz. Field D*: An interpolation-based path planner and replanner. In *Proceedings of the International Symposium on Robotics Research (ISRR)*, October 2005.
10. D. Frost, I. Rish, and L. Vila. Summarizing CSP Hardness with Continuous Probability Distributions. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence (AAAI'97/IAAI'97)*, pages 327–333. AAAI Press, 1997.
11. O. Hachour. Path planning of autonomous mobile robot. *International Journal of Systems, Applications, Engineering & Development*, 2(4):178–190, 2008.
12. P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics.*, 4:100?107, 1968.

13. H. Hoos and T. Stützle. Characterizing the Run-Time Behavior of Stochastic Local Search. In *Proceedings AAAI99*, 1998.
14. H. Hoos and T. Stützle. Towards a Characterisation of the Behaviour of Stochastic Local Search Algorithms for SAT. *Artificial Intelligence*, 112(1-2):213–232, 1999.
15. H. Hoos and T. Stützle. Local Search Algorithms for SAT: An Empirical Evaluation. *Journal of Automated Reasoning*, 24(4):421–481, 2000.
16. H. H. Hoos and T. Stützle. Evaluating Las Vegas Algorithms – Pitfalls and Remedies. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI-98)*, pages 238–245. Morgan Kaufmann Publishers, 1998.
17. I. Millington and J. Funge. *Artificial Intelligence for Games*. Morgan Kaufmann Publishers, 2 edition, 2009.
18. A. Nash, K. Daniel, S. Koenig, and A. Felner. Theta*: Any-angle path planning on grids. In *In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 1177–1183, 2007.
19. N. Nilsson. *Principles of Artificial Intelligence*. Tioga Publishing Company, Palo Alto, CA. ISBN 0-935382-01-1, 1980.
20. K. Sugihara and J. Smith. A genetic algorithm for 3-d path planning of a mobile robot. Technical report, Tech. Rep. No. 96-09-01. Software Engineering Research Laboratory,University of Hawaii at Manoa, 1996.