Sensitivity Analysis for Dynamic Control of PSTNs with Skewed Distributions

Rosy Chen[1*], Yiran Ma[2*], Siqi Wu[1*], James C. Boerkoel Jr.[1]

[1] Harvey Mudd College, Claremont, CA, USA
[2] Pomona College, Claremont, CA, USA
rochen@hmc.edu,ymac2020@mymail.pomona.edu, iwu@hmc.edu, boerkoel@hmc.edu

Abstract

Probabilistic Simple Temporal Networks (PSTN) facilitate solving many interesting scheduling problems by characterizing uncertain task durations with unbounded probabilistic distributions. However, most current approaches assess PSTN performance using normal or uniform distributions of temporal uncertainty. This paper explores how well such approaches extend to families of non-symmetric distributions shown to better represent the temporal uncertainty introduced by, e.g., human teammates by building new PSTN benchmarks. We also build probability-aware variations of current approaches that are more reactive to the shape of the underlying distributions. We empirically evaluate the original and modified approaches over well-established PSTN datasets. Our results demonstrate that alignment between the planning model and reality significantly impacts performance. While our ideas for augmenting existing algorithms to better account for human-style uncertainty yield only marginal gains, our results surprisingly demonstrate that existing methods handle positively-skewed temporal uncertainty better.

Introduction

Probabilistic Simple Temporal Networks (PSTNs) are a framework for modeling scheduling problems that include temporal uncertainty modeled as distributions. For example, PSTN can be used in human-robot teamwork settings to capture the uncertain tendencies exhibited by human autonomy. Existing experiments data on human temporal uncertainty conducted using an online collaborative human-robot packing game showed that heavy-tailed distributions, such as log-normal, best fit the task uncertainty introduced by humans in collaborative tasks (Dominguez, La, and Boerkoel 2020). This is corroborated by work that showed human reaction time is also best modeled as log-normal (Yin et al. 2013).

One shortcoming of most existing algorithms for solving PSTNs is that they have been evaluated exclusively on a narrow range of symmetric, well-behaved models of temporal uncertainty, such as Normal or Uniform distributions, which makes it unclear if they are well-suited for non-ordinary sources, such as the ones human teammates might introduce.

This paper addresses this gap by showing how well existing approaches work for solving PSTNs with non-symmetric uncertainty distributions. We provide methods to convert the distribution for existing benchmarks from uniform to heavy-tailed distributions and use the newly generated benchmarks to conduct a sensitivity analysis on existing algorithms for PSTNs under dynamic controllability. Then we present modified algorithms that utilize non-symmetry probabilistic information to account for the distributions we are interested in. In the end, we conclude with three main takeaways: (1) our ideas for augmenting existing algorithms to better account for human-style uncertainty only yield marginal differences, (2) alignment between the planning model and reality has a significant impact on performance, and (3) existing approaches generally control for temporal uncertainty better when it is positively skewed.

Background

A *Simple Temporal Network (STN)* is a graphical representation of scheduling problems. It consists of a set of timepoints, T = $t_0, t_1, t_2, \ldots, t_n$, each representing a temporal event, along with the set of temporal constraints between the timepoints C in the form of $t_j - t_i <= c_{ij}$ (Deichter, Meiri, and Pearl 1991). An STN schedule is a specific assignment of times to the events in a STN. The schedule is a solution if it satisfies all temporal constraints in the STN.

An *STN with Uncertainties (STNU)* extends the STN by partitioning timepoints into two types: a set of controllable events T^c, and a set of uncontrollable events T^u. The execution timepoints of controllable events T^c are subject to an agent's assignment, while the times for uncontrollable events in T^u are exogenously determined. Accordingly, two types of temporal constraints exist in an STNU. Contingent constraints are of the form $t_j - t_i \in [l_{ij}, u_{ij}]$ and the time elapses between is sampled from the range $[l_{ij}, u_{ij}]$ by an uncertain process. The rest are requirement constraints, which take the same form as constraints in regular STNs.

A *Probabilistic Simple Temporal Network (PSTN)* extends STNUs by allowing contingent edges between t_j and t_i to be determined by a random variable X_{ij} governed by a probability density function P_{ij} (Tsamardinos 2002). A *realization* is a selection of temporal values (e.g., sampled by Nature) for contingent edges in PSTNs. In a human-robot teamwork context, a realization is the timepoint a human chooses to execute an event, which is only observable to the dispatcher after it happens.

*These authors contributed equally.

Controllability

In STNUs and PSTNs, a schedule of controllable timepoints that guarantees the successful dispatch for every potential contingent outcome is called *strong controllability* (Vidal and Fargier 1999). *Dynamic controllability* is established for an STNU or PSTN if we can direct a scheduling strategy and ensure a successful dispatch using only knowledge of the timing of contingent timepoints that have already occurred. Dynamic controllability is much less constrictive and can be established in more cases than strongly controllable strategies. Since a PSTN can include unbounded distributions representing the contingent constraints, a controllable solution may not always exist.

Dispatch

Dispatch is deciding when an agent executes each of its timepoints. The basic dynamic dispatch algorithm we use is the early execution dispatch designed by Nilsson, Kvarnström, and Doherty (2014). This dispatcher works on controllable STNUs, and as shown by Akmal et al. (2019), can, but is not guaranteed to, work on uncontrollable ones. The strategy works by generating additional (wait) constraints and using them to help decide when a contingent timepoint can be executed (e.g., after all preceding events have been executed).

As a dynamic controllable dispatching strategy of PSTNs, *Min-Loss DC* algorithm (Gao, Popowski, and Boerkoel 2020), in two steps: First, it bounds every unbounded contingent edge in the original PSTNS by truncating an equal amount of probability from both sides. Second, the algorithm executes the DC-DISPATCH strategy if the STNU is DC. The Optimal DC Relaxation algorithm of Akmal et al. (2019) is applied when the bounded STNU is not DC. It returns an STNU that minimally squeezes the intervals of the contingent edges involved in the conflict.

Max-Gain is another algorithm that reduces a PSTN to a STNU (Gao, Popowski, and Boerkoel 2020). It cuts $\frac{\alpha}{2}$ probability from two ends of a contingent edge respectively, where α represents the risk budget. It tries to maximize the probability remaining on the cut contingent edges. It achieves so by conducting a binary search for the minimal α of all contingent edges that makes the cut STNU dynamically controllable. After finding the minimal working α for all contingent edges, it keeps reducing α to see if part of the contingent edges can regain more uncertainty. The contingent edges in the first appeared conflict is set back to the previous minimal α and excluded from the rest of the analysis, in which it keeps searching for a smaller α recursively on the rest of the network.

Sensitivity Analysis

To investigate the performance of current algorithms across representative types of distributions in human behavior, we conducted a sensitivity analysis specifically examined the successful scheduling rate using DC-Dispatch, Min-Loss, Max-Gain, and variations thereof on normal, gamma, exponential, lognormal, and beta benchmarks. We introduce modified benchmarks and algorithms to account for gamma, lognormal, exponential, and beta distributions.

We simulate sample realizations by sampling the distributions associated with each contingent edge. It is noted that the distributions we pass into Min-Loss and Max-Gain algorithms can be different from the realization distributions used in our simulation. Below we use *realization distribution* to represent, e.g., how a human behaves and *planning distribution* to describe the human behaviors predicted by the scheduling algorithm.

Benchmarks Variations

We adapt two existing PSTN benchmarks by replacing their normally-distributed uncertainty models with distributions that capture more natural sources of uncertainty. As a basis for our augmented benchmark, we use the DREAM benchmark of 540 PSTNs (Abrahams et al. 2019) and the CAR-SHARING dataset of 169 PSTNs which was created by Santana et al. (2016) and then edited by Akmal et al. (2019). [1]

Gamma benchmark has the shape parameter α set to proportional to the edge length. By doing so, this benchmark can represent the characteristics in human behavior that suggests tasks with longer duration tends to perform more normal distribution alike (Hyman 1953). The scale parameter β is set to one for consistency in comparison. The gamma benchmark tries to capture positively skewed heavy-tailed distributions.

Lognormal benchmark consists of lognormal distributions for each edge capturing 99% of probability density based on edge lengths. We chose the median value to be the midpoint of the edge and were able to attain μ. We then used inverse CDF to obtain the standard deviation σ and generated realizations based on the μ and σ. This benchmark represents distributions that are positively skewed which are more common in human behavior studies

Exponential benchmark aims to capture the most positively skewed distributions. The CDF of each edge is set to capture 99% of the probability density and gave the rate parameter λ accordingly.

Beta benchmark is symmetrical to the gamma benchmark to capture the negative-skewed distributions and extend our analysis to more general cases.

Beta continuous extends Beta benchmark by putting its parameters α and β on a continuous scale from 0.1 to 12 with a step size of 2. This way, we can use this distribution to model all the above distributions by setting the parameters to unique values. For example, $\alpha = 12$ and $\beta = 12$ approximate the normal distribution in the benchmark; $\alpha < \beta$ results in positively skewed distributions; and $\alpha > \beta$ results in negatively skewed distributions that can be explored. When $\alpha < 1 < \beta$, beta distribution decays exponentially. When $\beta < 1 < \alpha$, the distribution grows exponentially.

Algorithms Variations

We modify Min-Loss and Max-Gain to Min-Loss+ and Max-Gain+ so they adapt to handle probabilistic informa-

[1] Visit https://github.com/HEATlab/sensitivity/ for the augmented benchmarks and algorithm details w/ code.

tion for newly-added distributions.[1]

Min-Loss uses two phases to generate a controllable network. The first stage transforms boundless normal distributions to uncertain edges by evenly cutting $\frac{risk}{2}$ of probability on both ends. We vary the algorithm by extending the way of bounding edges based on probabilistic distributions. To lose as little uncertainty in terms of edge length, for example, for a positively-skewed distribution, we cut more probability from the left tail. We also explore alternatively leaving Min-Loss more room to resolve conflicts in the relaxation step by cutting more probability from the right tail instead.

The second stage of Min-Loss applies Optimal DC Relaxation on bounded contingent edges to attempt to generate dynamic controllable SNTUs. Conflicts are relaxed by shrinking the longer contingent edges by the same amount from the labeled directions, determined by the method in Bhargava, Vaquero, and Williams (2017). To extend Optimal DC Relaxation, we incorporate probabilistic density into the process and aim to minimize the loss of the expected uncertainty. The initial problem becomes a constraint optimization problem and can be solved by Lagrangian Multipliers (Bertsekas 1996). Across all long contingent edges involved in conflicts, we use Lagrangian multipliers to search for a local minimum sum of the product of the cut edge length and the areas enclosed by the cut and distribution function. This approach varies Min-Loss and allows it to account for non-symmetric distributions. Among the three variations, empirical results demonstrate only marginal differences in success rates. We focus on the best-performing algorithm—the one that biases toward removing less uncertainty in the first stage, which we refer to as Min-Loss+

We update the Max-Gain algorithm, which previously assumed PSTNs with normal distributions, to handle non-symmetric distributions. Cutting the same amount of probability from both ends of a non-symmetric distribution naturally results in different-sized cuts on the two ends. Further, in the particular case when the binary search approaches a risk budget of 1 ($\alpha \geq 0.999$), we operate as if the STNU edges were non-truncated rather than return an STNU where we truncate the intervals of contingent edges to a single point. We call this adapted algorithm Max-Gain+.

Empirical Evaluation

We evaluate the performance of DC-Dispatch, Min-Loss, Max-Gain, Min-Loss+, and Max-Gain+ on both the original DREAM and CAR-SHARING benchmarks, and also on our augmented benchmarks in which we incorporate gamma, exponential, lognormal, and beta distributions. We adopt the Monte Carlo sampling approach (Brooks et al. 2015) to assess algorithms' performance. The success rate of each algorithm is based on the number of generated valid schedules in every problem. For each instance, we run 200 trials and average the successful dispatching rate to approximate the performance for each pair of algorithms and benchmark. The experiment examines the performance of all modified methods and presents the best-performing modified algorithms, Min-Loss+ and Max-Gain+, for meaningful comparison. Specifically, both Min-Loss and Min-Loss+ used in this analysis have a risk level of 0.05.

methods	normal	gamma	exp	lognorm	beta
DC-Dispatch	0.38	0.46	0.51	0.33	0.06
Min-Loss	**0.49**	0.65	0.61	**0.42**	0.06
Max-Gain	**0.49**	**0.77**	**0.65**	0.36	**0.08**
Min-Loss+	0.49	0.65	0.61	**0.41**	0.06
Max-Gain+	**0.53**	**0.77**	**0.65**	0.39	**0.08**

Table 1: Performance of existing and modified dispatch strategies for CAR-SHARING benchmarks.

methods	normal	gamma	exp	lognorm	beta
DC-Dispatch	0.25	0.28	0.26	0.24	0.25
Min-Loss	**0.28**	**0.36**	**0.30**	**0.31**	0.12
Max-Gain	0.20	0.32	0.29	0.30	**0.31**
Min-Loss+	**0.28**	**0.36**	**0.30**	**0.32**	0.13
Max-Gain+	0.20	0.33	**0.30**	0.30	**0.31**

Table 2: Performance of existing and modified dispatch strategies for DREAM benchmarks.

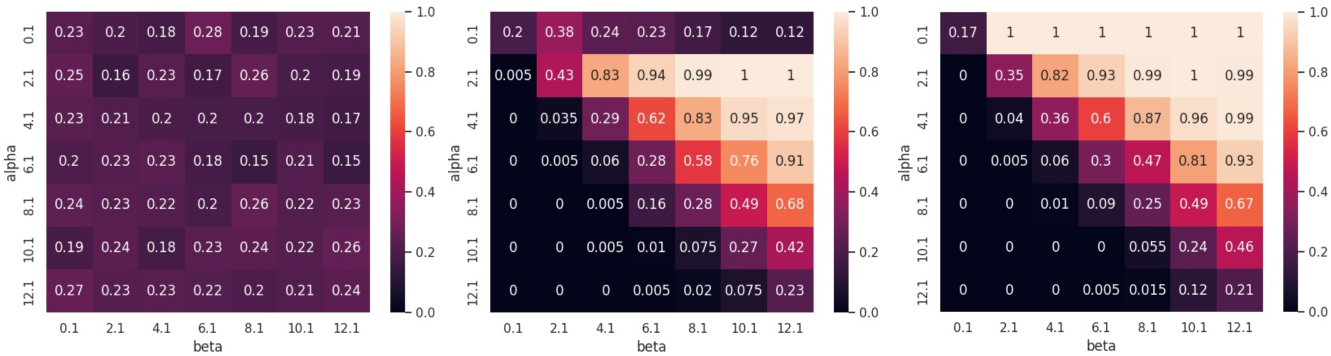
Empirical Results

Tables 1 and 2 confirms the prior conclusions that suggest Min-Loss has a better performance on symmetric distributions (Gao, Popowski, and Boerkoel 2020). At the same time, the across distributions result show that Min-Loss outperforms on all DREAM benchmarks except for beta distribution. While Max-Gain equals or outperforms on all CAR-SHARING benchmarks but lognormal distribution.

Tables 1 and 2 also demonstrate significant differences in all approaches' ability to optimize for various distributions. For instance, success rates for Max-Gain ranged from 0.77 on CAR-SHARING gamma benchmark to 0.08 on CAR-SHARING beta benchmark. Differences were much more muted on the DREAM benchmark, pointing to structural differences between the two benchmarks.

Finally, we notice only marginal differences exist between Min-Loss and Min-Loss+ and between Max-Gain and Max-Gain+ regardless of distribution types. In short, our modifications tended to shift how distributions were truncated but did so in the tails of the distributions, where the potential for gains is inherently smaller.

Uncertainty Model Alignment Next, we consider how much the alignment between planning and realization distributions matters. We performed a case analysis using a representative instance (test2) from CAR-SHARING dataset on the comprehensive beta-continuous benchmark and generated heat maps for success rates across combinations of $\alpha - \beta$ pairs. Figure 1 (a) has fixed realization distribution but varying planning distributions. Figure 1 (b) has varying realization distributions but fixed planning prediction. Figure 1 (c) uses the same realization distribution and planning distribution with changing $\alpha - \beta$ pairs. Figure 2 uses the same condition as Figure 1 (c) on the CAR-SHARING benchmark. To control variances introduced by different contingent edges in this benchmark, all contingent edges are set to have the same $\alpha - \beta$ pair, which differs from the other benchmarks, where contingent edges have the same trend,



(a) Normal Realization & Beta Planning (b) Beta Realization & Normal Planning (c) Matching (Beta) Realization & Planning

Figure 1: Performance across an instance while varying α and β of the Beta distribution

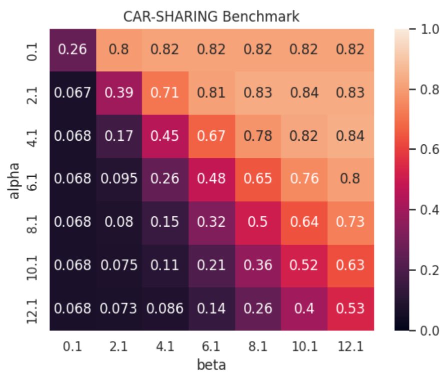


Figure 2: Matching Realization & Planning Distributions, CAR-SHARING Benchmark

but their $\alpha - \beta$ pairs may not be the same.

Figure 1 shows an asymmetry in the sensitivity of planning vs. reality misalignment. When an agent plans for heavy-tailed temporal uncertainty, but the reality is symmetric, it performs roughly equally well (or poorly), regardless of how heavy-tailed its uncertainty model was. On the other hand, when an agent plans for a symmetric, normal distribution, but the reality is heavy-tailed, the shape of that heavy-tailed distribution matters significantly. Interestingly, when an agent plans assuming a heavy-tailed distribution and matches reality, it performs pretty similarly as if it were to plan for a normal distribution—neither Figure 1(b) nor Figure 1(c) dominates the other.

More broadly, the shape of the realization distribution significantly impacts the success rate. In Figure 2, even though the realization and planning distributions matched, the ratio between α and β seems to significantly impact the agent's ability to plan effectively around the temporal uncertainty. Generally, as the ratio of β:α grows, the distribution becomes more positively skewed, and so does the algorithm's ability to exploit its knowledge of the uncertainty. This corroborates what we see in the earlier Tables, where all algorithms seemed better at exploiting positively skewed distributions such as the gamma or even exponential distributions.

Interestingly, algorithms that plan for positively skewed distributions start acting increasingly like very simple dispatch methods such as Next-First, which executes the next event it can as early as it can and has previously been shown to perform surprisingly well across many PSTN benchmarks (Saint-Guillain et al. 2021). Thus, this leads us to reject our original hypothesis that non-symmetric temporal uncertainty may require augmenting existing solution approaches. It may instead point to considering simple approaches such as Next-First when the nature of the temporal uncertainty skews positively, such as when dealing with human teammates.

Conclusion

This paper explores how well existing approaches handle non-symmetric distributions, for instance, in a human-robot team, where, from the robot's perspective, the human introduces temporal uncertainty. Like many temporal processes, the uncertain duration of human tasks tends to be positively skewed. We contribute new benchmarks that better capture many sources of real-world temporal uncertainty. This paper provides the first known look into how sensitive various approaches for controlling for temporal uncertainty within PSTN are and how sensitive these approaches are to the planning model of uncertainty aligning with reality. We show that our ideas for improving existing approaches to account for non-symmetric distributions had only a marginal impact, increasing the frontier of circumstances in which existing approaches have been shown to excel. Our results suggest there may be better ways to deal with natural sources of temporal uncertainty than further optimizing or engineering existing approaches. This corroborates previous results (Saint-Guillain et al. 2021) that showed simple approaches perform quite well on ordinary distributions derived from histograms of real-world tasks. In the future, we would like to explore whether there are domains where these more highly-optimized algorithms provide better trade-offs. This paper sheds light on future scheduling algorithm design, especially in domains where human temporal uncertainty significantly impacts outcomes, such as close human-robot collaborative tasks.

Acknowledgments

Funding for this work was graciously provided by the National Science Foundation under grant IIS-1651822. Thanks to the anonymous reviewers, HMC faculty, staff, and HEAT-lab members for their support and constructive feedback.

References

Abrahams, J. R.; Chu, D. A.; Diehl, G.; Knittel, M.; Lin, J.; Lloyd, W.; Boerkoel, J. C.; and Frank, J. 2019. DREAM: An Algorithm for Mitigating the Overhead of Robust Rescheduling. In *Proc. of the 29th International Conference on Automated Planning and Scheduling (ICAPS-19)*, 3–12.

Akmal, S.; Ammons, S.; Li, H.; and Boerkoel, J. C. 2019. Quantifying Degrees of Controllability in Temporal Networks with Uncertainty. In *Proc. of the 29th International Conference on Automated Planning and Scheduling (ICAPS-19)*, 22–30.

Bertsekas, D. P. 1996. *Constrained Optimization and Lagrange Multiplier Methods (Optimization and Neural Computation Series)*. Athena Scientific, 1 edition. ISBN 1886529043.

Bhargava, N.; Vaquero, T.; and Williams, B. 2017. Faster Conflict Generation for Dynamic Controllability. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, 4280–4286. AAAI Press. ISBN 9780999241103.

Brooks, J.; Reed, E.; Gruver, A.; and Boerkoel, J. C. 2015. Robustness in Probabilistic Temporal Planning. In *Proc. of the 29th National Conference on Artificial Intelligence (AAAI-15)*, 3239–3246.

Dominguez, M. A.; La, W.; and Boerkoel, J. C. 2020. Modeling Human Temporal Uncertainty in Human-Agent Teams. *CoRR*, abs/2010.04849.

Gao, M.; Popowski, L.; and Boerkoel, J. 2020. Dynamic Control of Probabilistic Simple Temporal Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(06): 9851–9858.

Hyman, R. 1953. Stimulus information as a determinant of reaction time. *Journal of experimental psychology*, 45 3: 188–96.

Nilsson, M.; Kvarnström, J.; and Doherty, P. 2014. Classical Dynamic Controllability Revisited - A Tighter Bound on the Classical Algorithm. In *Proc. of 6th International Conference on Agents and Artificial Intelligence (ICAART-14), 6-8 March 2014, Angers, France*, 130–141.

Saint-Guillain, M.; Vaquero, T.; Chien, S.; Agrawal, J.; and Abrahams, J. 2021. Probabilistic Temporal Networks with Ordinary Distributions: Theory, Robustness and Expected Utility. *Journal of Artificial Intelligence Research*, 71: 1091–1136.

Santana, P.; Vaquero, T.; Toledo, C.; Wang, A.; Fang, C.; and Williams, B. 2016. Paris: a polynomial-time, risk-sensitive scheduling algorithm for probabilistic simple temporal networks with uncertainty. In *Proc. of the 26th International Conference on Automated Planning and Scheduling (ICAPS-16)*, 267–275.

Tsamardinos, I. 2002. A Probabilistic Approach to Robust Execution of Temporal Plans with Uncertainty. In *Methods and Applications of Artificial Intelligence*, 97–108.

Vidal, T.; and Fargier, H. 1999. Handling Contingency in Temporal Constraint Networks: from Consistency to Controllabilities. *Journal of Experimental and Theoretical Artificial Intelligence*, 11(1): 23–45.

Yin, P.; Luo, P.; Lee, W.-C.; and Wang, M. 2013. Silence is Also Evidence: Interpreting Dwell Time for Recommendation from Psychological Perspective. In *Proc. of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '13, 989–997.