Models of Robustness for Temporal Planning and Scheduling with Dynamic Controllability

Jing Cui
Optimisation Research Group, NICTA
Research School of Computer Science, ANU
jing.cui@nicta.com.au

Abstract

Temporal uncertainty is a basic feature in many real scheduling and planning problems. How to achieve a robust schedule or plan which can deal with the temporal uncertainty is a useful issue in real problems. Current research on this issue has introduced different robustness measures. Although they all reflect the quality of temporal schedules or plans in different perspectives, the lack of research on the relations and deviations among the measures is a problem. The plan of my research is to explore a variety of robustness metrics, to find the relations and differences among them and to figure out the model of robust model about schedules and temporal plans. Finally, the goal of my work is to find a better way to explain robustness in scheduling and temporal planning and use the robustness metrics to achieve robust schedules and temporal plans.

Motivation

Different kinds of uncertainty exist in real scheduling problems, e.g., train scheduling with possible delays caused by natural or human factors, personal travel planning with fluctuating public transportation timetables and manufacture process scheduling with possibly not working machine. How to generate robust schedules according to these uncertain situations has been considered in many research (Banerjee and Haslum 2011; Policella et al. 2004; 2007). But before considering approaches to solve such uncertain scheduling problems, the question of what robustness is needs to be answered first.

Robustness is hard to measure because it is an abstract concept. However, it is easier to evaluate specific features related to robustness. And to a certain extent, measuring robustness-related features can reflect the quality of schedules. Each measure indicates how good the schedule is in its own view. These views may consider different features related to robustness such as flexibility, size, stability, etc.. Thus, comparing the same pair of schedules, we may get different answers to the question, which schedule is more robust? The goal of my work is to explore a variety of robustness metrics, to figure out what exactly these metrics measure, whether there is any relationship among them, and even to find out a better way to measure the robustness of schedules and temporal plans.

Background

Before discussing the measure of schedules or temporal plans, we will review the current temporal reasoning model first. Then we will review several different metrics of schedules.

Models of Schedule and Temporal Plan

Dechter, Meiri, and Pearl (1991) introduced the Simple Temporal Networks (STN) which is a general way to represent temporal problems. In this subsection, we will review two main extensions of the original STN.

STN with Uncertainty (STNU) Formally, an STNU (Vidal and Fargier 1999) is an extension of STN which consists of a set of nodes $X = X_E \cup X_U$, representing executable (X_E) and uncontrollable (X_U) time points, and a set of links $E = R \cup C$, called *requirement* and *contingent* links. Each link e_{ij} has a lower bound L_{ij} and upper bound U_{ij}, representing the constraints $L_{ij} \leq t_j - t_i \leq U_{ij}$. Each uncontrollable time point has exactly one incoming contingent link, whose lower bound is non-negative. In other words, executable time points correspond to choices of the agent, contingent links represent uncontrollable durations, and the uncontrollable time points are when the agent finds out what duration the environment has chosen. Requirement links may connect any pair of time points.

Because of the uncertainty feature in STNU, consistency and controllability are vital (Vidal and Fargier 1999). We will discuss the controllability of STNU in the next section.

Probabilistic STN (pSTN) It is a natural extension to the STNU model to associate probabilities with the outcome (timing) of uncontrollable events (Tsamardinos 2002; Fang, Yu, and Williams 2014). The pSTN redefines the representations of the contingent links. Instead of the lower and upper bounds of the links, pSTN uses probabilistic distributions to show the temporal uncertainty. The **uncertain duration** $D_{ij} = t_j - t_i : \Omega \to R$ is a random variable describing the duration of the contingent link.

From the probabilistic STN, we can measure the robustness of schedules and temporal plans by calculating the possibility of success. Furthermore, the possibility of failure can be treated as constraint and other more important preferences could be the objectives (Fang, Yu, and Williams 2014).

Robust Measures on Schedules

In this section, we will review four existing robustness measures on Partial Order Schedule (POS). A POS is a consistent STN, which is a partial solution of scheduling problem with resource constraints. By adding additional temporal constraints, POS resolve all the resource constraints and maintain the original temporal constraints from the problem. So a POS represents a set of solutions of the scheduling problem. And there may be different POS for one problem. The problem that which POS is the most robust one or more robust than some other ones is important.

Flexibility This metric measures the flexibility of Partial Order Schedules (POS) (Aloulou and Portmann 2003).

However, different POS may be got from the same problem. This metric – flexibility – helps to decide which one is better. To be more precise, this metric counts the number of pairs that do not have any explicit or implicit precedence relations in a POS. The definition of $flex$ is

$$flex = \frac{|\{(a_i, a_j)|a_i \nprec a_j \wedge a_j \nprec a_i\}|}{n(n-1)} \tag{1}$$

where the precedence constraints between activities include both explicit and implicit relations. Thus, the higher $flex$ the POS gets, the lower degree of interaction among activities it achieve and the more solutions it represents.

Even though $flex$ implies robustness-related features, such as independence, it does not consider temporal slacks or redundancy in the POS. If a POS has low $flex$ but every time constraint is loose enough to allow a certain range of diviation, $flex$ is not able to show its advantages.

Fluidity In order to take slacks into account when measuring robustness, Cesta, Oddi, and Smith (1998) introduced fluidity. It represents the ability to absorb temporal deviation. It is defined as

$$fldt = \sum_{i=1}^{n} \sum_{j=1 \wedge j \neq i}^{n} \frac{slack(a_i, a_j)}{H \times n \times (n-1)} \times 100 \tag{2}$$

where H is a fair bound which is large enough to allow all activities to be executed, and $slack(a_i, a_j)$ is the width of the allowed distance interval between two activities. The higher the $fldt$ is, the lower the risk of cascading change, the higher tolerance to temporal deviation, and the higher the probability of changes in response to disruption remaining local.

Improved Fluidity Wilson et al. (2014) proposed an improved metric which measure the total temporal tolerance of an STN. The authors thought the previous $fldt$ overestimated the robustness by calculating dependent slacks repeatedly. For instance, a sequence of activities need to be done one by one and the total time should be within a certain upper bound. Then the temporal slack of the whole sequence will be calculated as many times as the number of activities as the original fluidity metric.

The improved fluidity metric treats every activity as two separated nodes which are a start and an end nodes and every original constraint as a end-to-start constraint. The fluidity is optimising the sum of every slack between the start and end nodes of one activity, which satisfies all the temporal constraints.

By solving the following LP model, we can get the improved fluidity.

$$\begin{aligned} max \ & \sum_{t \in T} (end(t) - start(t)) \\ s.j. \ & start(t) \leq end(t) \qquad t \in T \\ & start(t_i) - end(t_j) \leq c \quad \forall (t_i - t_j \leq c) \in C \end{aligned} \tag{3}$$

In the formula, T and C are the sets of activities and constraints in the original STN separately.

This metric improves the original fluidity by considering dependency of the activities. And it can be extended to STNU as well. Wilson et al. introduce a model to calculate the fluidity of a STNU subject to strongly controllable constraints.

Disruptibility The former metrics measure robustness in terms of the flexibility of a POS. This measure called disruptibility was introduced by Policella et al. (2004). It considers stability against changes. The definition of $dsrp$ is

$$dsrp = \frac{1}{n} \sum_{i=1}^{n} \frac{slack(a_i)}{num_{changes}(a_i, \Delta a_i)} \tag{4}$$

The $slack$ here is different from the one used in $fldt$. It represents the temporal variability of a single activity and equals the difference between the upper bound and the lower bound of the end time of activity a_i. The function $num_{changes}(a_i, \Delta a_i)$ counts the number of activities which are changed in the process of right-shifting activity a_i by an amount of time Δa_i. So this metric takes disruption into account and calculates the influence of temporal delay of activities. The essence of this measure estimates the trade-off between flexibility and the implied changes.

Summary Different robustness measures evaluate the quality of a schedules from different views. However, whether the "good" solution appraised by each measure is really good and whether a better solution according to one measure will still be the better one according to other measures, are not guaranteed. For instance, a POS with too much flexibility will not achieve a high value of disruptibility and a POS with high fluidity may get low flexibility as well. Furthermore, regardless of what specific features those measures exactly take into account, they calculate the average level across the schedule, which means they ignore the deviation in detail. Specifically, if there is a *weak point* in the POS such as a link with a tight time constraint or a small group of activities with strong precedence constraints, then it is still possible to achieve a good average value when the rest of the schedule is strong enough. Therefore, we need a deep view of the relationships and differences among the metrics. And a better model to calculate robustness of schedules and temporal plans is worth exploring.

Robustness Measures with Dynamic Controllability

In this section, we introduce the model of robustness measures of schedules that is dynamically controllable. The basic idea of the model is to optimise robustness of an STNU with satisfying the constraints of dynamic controllability. The robustness in the model is represented by the objective function f of the optimising problem.

We will use the following notation: For each link e_{ij}, l_{ij} and u_{ij} are the decision variables, representing the lower and upper bounds on this link. These are contained by constant outer bounds $L_{ij} \leq l_{ij} \leq u_{ij} \leq U_{ij}$. If, for a particular link, no outer bounds are available, we set $L_{ij} = -\infty$, $U_{ij} = \infty$. However, in many of the application problems we consider (cf. next section) tight outer bounds are given for most links, which allows the constraint model to be simplified.

$$\begin{array}{lll} \min & f(l_{ij}, u_{ij}) & e_{ij} \in R \cup C \\ \text{s.t.} & L_{ij} \leq l_{ij} \leq u_{ij} \leq U_{ij} & e_{ij} \in R \cup C \\ & \text{dynamic controllability} & \\ & \text{other constraints for contingent links} & \end{array}$$

Here, R and C are the sets of requirement and contingent links separately. And *other constraints for contingent links* mean different models for the contingent links could be used according to different backgrounds of the problems.

Dynamic Controllability (DC)

In an STN (without uncertainty), the tightest bounds on the difference between any two time points that are implied by the given links can be computed in polynomial time by an all-pairs shortest path algorithm. Thus, there is an implicit requirement link between every pair of time points. If any link's bounds are inconsistent ($L_{ij} > U_{ij}$), the network has no satisfying assignment (Dechter, Meiri, and Pearl 1991).

Checking dynamic controllability of an STNU was first shown to be tractable by Morris, Muscettola and Vidal (2001). Their algorithm repeatedly applies a set of reductions, tightening the bounds on requirement links, until no more reductions apply (in which case the network is controllable) or the network becomes inconsistent (implying it is not controllable).

Morris, Muscettola and Vidal's (2001) algorithm repeatedly examines each triangle of time points in the STNU, considering at most one contingent link each time. Figure 1 shows a triangle, with time points A, B and C. The link between A and C is contingent, the other two are requirement links. (If e_{AB} is also contingent, it is considered in a separate triangle. Recall that there is an implicit requirement link between each pair of time points, including parallel to the contingent links.) First, it applies the implied (shortest path) bounds (e.g., $L_{BC} \leftarrow \max(L_{BC}, L_{AC} - U_{AB})$ and $U_{BC} \leftarrow \min(U_{BC}, U_{AC} - L_{AB})$). The next step depends on the relation between B and C:

• If $U_{BC} < 0$, B must be scheduled after C (hence, after C has been observed), so no further adjustments are needed. This is called the "follow" case.

• If $L_{BC} \geq 0$, B must be scheduled before or simultaneously with C (i.e., before C has been observed). This is called the "precede" case, and the bounds on the e_{AB} link are updated to $L_{AB} \leftarrow \max(L_{AB}, U_{AC} - U_{BC})$ and $U_{AB} \leftarrow \min(U_{AB}, L_{AC} - L_{BC})$.

• If $L_{BC} < 0$ and $U_{BC} \geq 0$, B may be scheduled before or after C. This case, called the "unordered" case, introduces a conditional bound, called a "wait", $\langle C, U_{AC} - U_{BC} \rangle$, with the meaning that execution of B must wait for either C to occur or at least $U_{AC} - U_{BC}$ after A. If $U_{AC} - U_{BC} \leq L_{AC}$, C cannot occur before the wait expires, so the wait is replaced by the unconditional bound $L_{AB} \leftarrow \max(L_{AB}, U_{AC} - U_{BC})$. Otherwise, $L_{AB} \leftarrow \max(L_{AB}, L_{AC})$, since the wait for C will delay B to at least L_{AC} after A.

Tighter bounds on a requirement link propagate to any other triangle that the link is part of. In addition, the algorithm performs "wait regression", in which conditional bounds are propagated to other links. If $\langle C, w \rangle$ is a wait on e_{AB}, where $w \leq U_{AC}$, then (i) if there is any link e_{DB} with upper bound U_{DB}, a wait $\langle C, w - U_{DB} \rangle$ is added to e_{AD}; and (ii) if $w \geq 0$ and there is a contingent link e_{DB}, where $B \neq C$, with lower bound L_{DB}, a wait $\langle C, w - L_{DB} \rangle$ is added to e_{AD}.

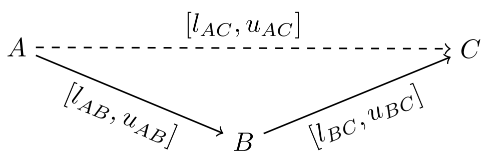

Figure 1: An STNU triangle. The A–C link is contingent.

Constraint Model of DC

The constraint model uses essentially the same reduction rules, but in the form of constraints between the decision variables that represent link bounds. Constraints are formulated over each triangle of time points in the STNU, considering at most one contingent link each time.

Shortest path constraints

$$\begin{array}{rcccl} l_{AC} & \leq & u_{AB} + l_{BC} & \leq & u_{AC} \\ l_{AC} & \leq & l_{AB} + u_{BC} & \leq & u_{AC} \\ u_{AC} & \leq & u_{AB} + u_{BC} & & \\ & & l_{AB} + l_{BC} & \leq & l_{AC} \end{array} \quad (5)$$

The shortest path constraints can propagate in any direction (i.e., from the contingent link e_{AC} to the requirement links and vice versa.) This may seem contradictory, since a contingent link may not be squeezed by requirements. However, l_{AC} and u_{AC} here are not given bounds but decision variables, whose final values will be the bounds on the contingent link. In some applications (e.g., the problem of minimising flexibility which motivated Wah and Xin) these variables are fixed to the given outer bounds (by adding constraints $l_{AC} = L_{AC}$, $u_{AC} = U_{AC}$) but other applications allow the bounds of contingent links to vary.

If no link in the triangle is contingent, these are the only constraints. Assuming, w.l.o.g., that the e_{AC} link is contingent, what constraints are needed is determined by the outer bounds on the e_{BC} link, following the cases in Morris, Muscettola and Vidal's (2001) algorithm. If $U_{BC} < 0$, then

$u_{BC} < 0$ and the triangle must always be in the follow case. Thus, no additional constraints are needed.

Precede constraints If $L_{BC} \geq 0$, then $l_{BC} \geq 0$ and the triangle will be in the precede case. The following constraints must hold:

$$\begin{aligned} u_{AB} &\leq l_{AC} - l_{BC} \\ l_{AB} &\geq u_{AC} - u_{BC} \end{aligned} \tag{6}$$

This together with (5) is equivalent to

$$\begin{aligned} u_{AB} &= l_{AC} - l_{BC} \\ l_{AB} &= u_{AC} - u_{BC} \end{aligned} \tag{6'}$$

since $l_{AB} \leq u_{AB}$ is always required. If $L_{BC} < 0$ and $U_{BC} \geq 0$, the triangle can be in any case, depending on the values given to l_{BC} and u_{BC}. The precede constraint then becomes disjunctive:

$$(l_{BC} < 0) \ \vee \ \left(\begin{aligned} u_{AB} &\leq l_{AC} - l_{BC} \\ l_{AB} &\geq u_{AC} - u_{BC} \end{aligned} \right) \tag{7}$$

Triangular wait constraints If it is possible that the triangle may be in the unordered case ($L_{BC} < 0$ and $U_{BC} \geq 0$), a variable representing the conditional wait bound is added:

$$w_{ABC} \ \geq \ u_{AC} - u_{BC} \tag{8}$$

Regression of waits (described below) may introduce wait variables w_{ABX}, where X is any uncontrollable time point (not necessarily in the same triangle as A and B). For each requirement link e_{AB} and wait variable w_{ABX}, the following disjunctive constraint must hold:

$$l_{AB} \ \geq \ \min(l_{AX}, w_{ABX}) \tag{9}$$

If $U_{AB} \leq L_{AX}$, this simplifies to $w_{ABX} = l_{AB}$. The constraint $w_{ABX} \leq u_{AB}$ must also hold.

Wait regression Each wait bound $\langle X, t \rangle$ on a link e_{AB} is represented by a variable w_{ABX}. However, as explained in the next section, many of these constraints are redundant and can be left out.

If there is a wait w_{ABX} and a contingent link e_{DB}, then wait regression implies the constraint

$$(w_{ABX} < 0) \ \vee \ (\ w_{ADX} \ \geq \ w_{ABX} - l_{DB}\ ) \tag{10}$$

It is disjunctive because the regression only applies when $w_{ABX} \geq 0$. If e_{DB} is a requirement link, only the weaker constraint

$$w_{ADX} \ \geq \ w_{ABX} - u_{DB} \tag{11}$$

must hold.

Formulation as a MIP

The disjunctions in constraints (7) and (9) mean the model is not a linear program. Wah and Xin (2004) used non-linear constraints to encode the disjunctions (as explained in the next section) and tackled it with the non-linear programming solver SNOPT. As an alternative, we formulate a mixed-integer linear programming (MIP) formulation, where disjuctions are encoded using binary (0/1) variables. Although MIP is an NP-hard problem, MIP solvers such as CPLEX or Gurobi are often very efficient in practice, and, in particular, typically more efficient than non-linear solvers. Experiment results across all application problems confirm this.

The wait bound constraint (9) can be formulated as

if $\alpha > 0$ then $\beta \geq 0$ else $\gamma \geq 0$

where $\alpha = w_{ABX} - l_{AX}$, $\beta = l_{AB} - l_{AX}$ and $\gamma = l_{AB} - w_{ABX}$ are all linear expressions. The disjunction can be replaced by the following linear constraints

$$\alpha - xU_\alpha \ \leq \ 0 \tag{12a}$$

$$\alpha - (1-x)(L_\alpha - 1) \ > \ 0 \tag{12b}$$

$$\beta - (1-x)L_\beta \ \geq \ 0 \tag{12c}$$

$$\gamma - xL_\gamma \ \geq \ 0 \tag{12d}$$

where $x \in \{0, 1\}$ is a binary variable, L_α, L_β and L_γ are constant lower bounds on α, β and γ, respectively, and U_α is a constant upper bound on α. This forces $\alpha > 0$ and $\beta \geq 0$ when $x = 1$, and $\alpha \leq 0$ and $\gamma \geq 0$ when $x = 0$. In the wait bound constraint, where $\alpha = w_{ABX} - l_{AX}$, we can choose $U_\alpha = U_{AB} - L_{AC}$, because U_{AB} is an upper bound on w_{ABX} ($w_{ABX} \leq u_{AB} \leq U_{AB}$) and L_{AC} is a lower bound on l_{AC}. The wait w_{ABX} is lower-bounded by the maximum of all wait constraints – triangular and regressed – on e_{AB}. The triangular wait lower bound is $w^t_{ABX} = u_{AX} - u_{BX}$, and from the shortest path constraint $l_{AX} + u_{BX} \leq u_{AX}$ we have $w^t_{ABX} \geq l_{AB}$. Thus, we can choose $L_\alpha = L_{AB} - U_{AC}$. For the lower bounds on $\beta = l_{AB} - l_{AX}$ and $\gamma = l_{AB} - w_{ABX}$ we have $L_\beta = L_{AB} - U_{AC}$ and $L_\gamma = L_{AB} - U_{AB}$, respectively.

The precede constraint (7) and regressed wait bound (10) are similar, except they have conditions only in one of the two cases (either "then" or "else"). Where one side of a disjunction consists of (a conjunction of) several linear constraints, as in (7), it is only necessary to add a constraint like (12c) for each conjunct, all using the same binary variable.

The wait regression constraint (10) can be strengthened to

$$(w_{ABC} \leq l_{AC}) \ \vee \ (w_{ADC} \geq w_{ABC} - l_{DB}) \tag{10'}$$

The advantage of this is that one disjunct, $w_{ABC} \leq l_{AC}$, is the same as the branching condition in (9), so both constraints can be captured with one binary variable.

Constraint (10') is valid because regressing a wait w_{ABC} through a contingent link e_{DB}, via (10), is redundant if w_{ABC} is not greater than the lower bound of the contingent link e_{AC} that causes the wait. If $w_{ABC} \leq l_{AC}$, the wait bound constraint (9) implies $l_{AB} \geq w_{ABC} \geq 0$. Hence, the triangle DAB is in the precede case and $l_{AD} = -u_{DA} = -(l_{DB} - l_{AB}) = w_{ABC} - l_{DB}$, which implies the wait regression constraint (10).

Formulation as an NLP

The non-linear model formulated by Wah and Xin (2004) uses quadratic constraints and terms in to the objective function to encode disjunctions. The precede constraint (7) is formulated as follows:

$$l_{BC}(l_{AB} - u_{AC} + u_{BC}) \ \geq \ 0 \tag{13a}$$

$$l_{BC}(u_{AB} - l_{AC} + l_{BC}) \ \leq \ 0 \tag{13b}$$

For each wait bound constraint (9), they introduce an auxiliary variable β and the following constraints:

$$(l_{AX} - w_{ABX})(l_{AB} - w_{ABX}) \geq 0 \quad (14a)$$
$$\beta \geq 0 \quad (14b)$$
$$\beta \geq w_{ABX} - l_{AX} \quad (14c)$$
$$\beta(l_{AB} - l_{AX}) \geq 0 \quad (14d)$$

Furthermore, a quadratic term $\beta(\beta - (w_{ABX} - l_{AX}))$ is added to the objective function. (This term must be minimised; if the problem is one of maximisation, its negative is used.) Its purpose is to ensure that β is 0 when $w_{ABX} \leq l_{AX}$ and otherwise equal to the difference $w_{ABX} - l_{AX}$. For this to work, the penalty incurred by a non-zero value of this term outweigh the actual objective function. Note also that there is a situation in which this formulation fails to impose the lower bound on l_{AB}, since if $l_{AX} = w_{ABX}$, constraint (14a) is satisfied even if $l_{AB} \not\geq w_{ABX}$, and (14b)–(14d) are satisfied by setting $\beta = 0$. A similar encoding is used for the wait regression constraint (10').

Applications

In this section, we will introduce several applications that are specific measures of robustness on schedules based on dynamic controllability. The first measure is a non-probabilistic measure, which regards the maximum temporal deviations of the activities as the robustness metric. And the second measure is a probabilistic one, which optimise the probability of success as the robustness metric. At last, we also compare dynamic controllability with strong control, which shows that dyanmic control enlarge the domain of solutions which may be helpful in real scheduling problems.

Robustness with Non-Probabilistic Uncertainty (Maximum Deviation)

In abstract terms we may define robustness as the greatest level of disturbance (deviation from expected outcomes) at which the schedule is still successfully executed. To operationalise this definition, we have to specify what kind of disturbances are considered, and how the schedule executive can use flexibility to cope with them. Here, we exemplify by assuming (1) that the possible disturbances are deviations in the time taken to execute an activity from its normal duration, and (2) a partial-order schedule with a dynamic execution strategy.

A partial-order schedule (POS) consists of a set of time constraints between activities such that any realisation that meets these constraints is also resource feasible. In the deterministic case, where the duration of each activity i is a constant d_i, the POS can be represented as an STN with time points t_{s_i} and t_{e_i} for the start and end, respectively, of each activity. Assuming the duration of each activity can vary within some bounds, $[l_{s_i,e_i}, u_{s_i,e_i}]$, the schedule can be modelled as an STNU where the link $e_{s_ie_i}$ from each activity's start to its ending time point is contingent, while remaining time constraints are requirement links. Thus, given a POS we can ask, what is the maximum deviation (i.e., width of the contingent bound) on any activity at which the STNU is dynamically controllable? This defines our measure of robustness. To compute it, we solve the following problem:

$$\begin{array}{lll} \max & \Delta & \\ \text{s.t.} & l_{s_i,e_i} = d_i - \delta_i \geq 0 & \forall i \\ & u_{s_i,e_i} = d_i + \delta_i & \forall i \\ & 0 \leq \Delta \leq \delta_i & \forall i \\ & \text{POS constraints (requirement links)} & \\ & \text{dynamic controllability (5)–(11)} & \end{array}$$

As explained above, requirement link bounds must be allowed to shrink, but their outer bounds can be set to the given POS constraints. Since contingent links represent durations, a hard lower bound $L_{s_ie_i} = 0$ applies.

We can also define a one-sided variant of this robustness metric, accounting for delays only, by fixing $l_{s_i,e_i} = d_i$ (i.e., adding deviations only to the upper bound).

Robustness with Probabilistic Uncertainty (Minimum Risk)

It is a natural extension to the STNU model to associate probabilities with the outcome (timing) of uncontrollable events (Tsamardinos 2002; Fang, Yu, and Williams 2014). As mentioned earlier, this allows us to define robustness as the probability of successful plan or schedule execution.

In the probabilistic STN (pSTN) model proposed by Fang, Yu and Williams (2014) the duration of each contingent link e_{ij} (i.e., the difference $t_j - t_i$) is a random variable D_{ij}. The model makes no assumption about independence or the distribution of these variables.

It is straightforward to transfer the STNU representation of a partial-order schedule, described above, to a pSTN. Since the random variable $D_{s_ie_i}$ represents the duration of an activity it is reasonable to assume it to be non-negative. The probability of a successful dynamic schedule execution is then at least

$$\begin{array}{lll} \max & \mathbf{P}\left(\bigwedge_i l_{s_ie_i} \leq D_{s_ie_i} \leq u_{s_ie_i}\right) & \\ \text{s.t.} & 0 \leq l_{s_i,e_i} \leq u_{s_i,e_i} & \forall i \\ & \text{POS constraints (requirement links)} & \\ & \text{dynamic controllability (5)–(11)} & \end{array}$$

The objective value is a conservative (lower) bound on the success probability, because it is the probability that all uncontrollable events fall inside the chosen bounds. The schedule may still execute successfully even if some durations fall outside these bounds, as the outcomes of other events may fortuitously compensate so that no constraints are violated.

The form of the objective function depends on the probability distributions over activity durations. If, for example, each $D_{s_ie_i}$ is uniform over an interval $[L_{s_ie_i}, U_{s_ie_i}]$, the probability of each uncertain duration falling within its bounds equals the proportion of the interval covered, i.e., $\mathbf{P}(l_{s_ie_i} \leq D_{s_ie_i} \leq u_{s_ie_i}) = \frac{u_{s_ie_i} - l_{s_ie_i}}{U_{s_ie_i} - L_{s_ie_i}}$. Applying the union bound (Boole's inequality), $\mathbf{P}\left(\bigvee_i (D_{s_ie_i} \leq l_{s_ie_i} \vee D_{s_ie_i} \geq u_{s_ie_i})\right) \leq \sum_i \mathbf{P}\left(D_{s_ie_i} \leq l_{s_ie_i} \vee D_{s_ie_i} \geq u_{s_ie_i}\right)$ a still (more) conservative estimate of the success probability can be obtained with a linear objective function. Other distributions give rise to a non-linear objective function, which we can only solve in combination with the non-linear formulation of the dynamic controllability constraints.

Dynamic vs. Strong Controllability

We compare the quality of solutions that can be obtained under dynamic and strong controllability constraints on the test cases used by Fang, Yu and Williams (2014). The authors argue that in many situations, minimising risk (probability of failure) is too conservative, and may compromise other objectives (such as cost) too much. Instead, users may prefer to give an absolute bound on risk and optimise other metrics subject to this constraint. They propose a pSTN optimisation algorithm subject to strong controllability and the chance constraint

$$\sum_{e_{ij} \in C} (1 - \mathbf{P}(l_{ij} \le D_{ij} \le u_{ij})) \ \le \ \rho. \qquad (15)$$

(This again makes use of the union bound, and so is conservative in both senses described above. That is, it ensures the probability of violating a requirement is ρ *or less*.)

Combining the dynamic controllability model (5)–(11) with (15) enables us to find dynamic execution strategies under chance constraints. Whether the constraint is linear or non-linear, and hence which solvers can be applied, depends on the probability distributions, as noted above.

Since these problems feature normally distributed uncertain durations, only the non-linear solver is able to tackle them. We therefore use the non-linear constraint model. The objective is minimising makespan.

Since a strongly controllable network is also dynamically controllable, the objective value can only improve (i.e., makespan decrease). Figure 2 shows the distribution of the improvement achieved under dynamic controllability over that achievable under strong controllability. (Results with strong controllability are generated by Fang, Yu and Williams' solver.) The improvement is measured by the reduction in makespan from that of the strongly controllable solution, expressed as a fraction of the latter. This clearly shows the value of using a dynamic execution strategy. (In fact, the possible improvement may be even greater since solutions returned by the non-linear solver are often not optimal.) 8.5% of problems are infeasible under strong controllability constraints – that is, no strong (unconditional) execution strategy exists – but are feasible under dynamic controllability constraints.

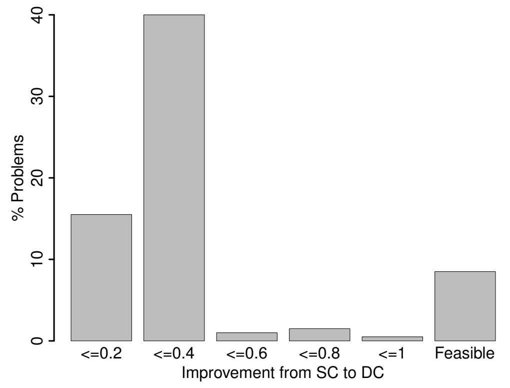

Figure 2: Reduction in makespan achieved with dynamic as opposed to strong controllability. Instances in the last column are infeasible under strong controllability, but have a valid dynamic execution strategy.

Summary and Future Work

Exploring the metrics of robustness with dynamic controllability can reflect the quality of schedules and it is worth doing. But to solve such an optimising problem is not easy. The constraint model of dynamic controllability we considered follows closely the algorithm by Morris, Muscettola and Vidal (2001). It is an open question if a different, perhaps more easily solved, model can be derived from the Morris' (2006) structural characterisation of dynamic controllability. Another possible method to improve the performance is to try other solvers, for instance as meta-CSP which is used to solve the disjunctive temporal problems (Tsamardinos and Pollack 2003; Moffitt 2011).

Besides the work to improve the current experiments, the future plan of my research has some possible directions. The first direction is combining the robustness measure based on dynamic controllability and current algorithms (Banerjee and Haslum 2011) to solve scheduling problems. Instead of just measuring the quality of the results, we can find dynamically controllable schedules with high objective value. When a schedule is dyanmically controllable, it can be executed successfully if the agent could always observe the past and make decision for the future (Vidal and Fargier 1999). Therefore, the results is far more useful than simply judging the quality of the schedules. However, there is an obvious difficulty in this approach, which is the run time of solving an optimising problem with constraints of dynamic controllability. So the improvements of the current work is necessary.

Another possible direction is to step from measures of scheduling towards that of temporal planning. Temporal planning and scheduling have interdependencies both in approaches and representations (Smith, Frank, and Jonsson 2000). The robustness measure of scheduling can reflect the quality of temporal plan to some extent. But planning focuses on what activities should be done, which contains more factors that could influence the robustness of the plan.

Last but not least, besides the aim to achieving robust results, it is also worth optimising other preferences (i.e., minimising cost) with a certain range of robustness.

References

Aloulou, M. A., and Portmann, M.-C. 2003. An efficient proactive reactive scheduling approach to hedge against shop floor disturbances. In *Proc. 1st Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA)*, 337–362.

Banerjee, D., and Haslum, P. 2011. Partial-order support-link scheduling. In *Proc. 21st International Conference on Automated Planning and Scheduling (ICAPS)*, 307–310.

Cesta, A.; Oddi, A.; and Smith, S. F. 1998. Profile-based algorithms to solve multiple capacitated metric scheduling problems. In *Proc. 4th International Conference on Artificial Intelligence Planning and Scheduling (AIPS)*, 214–223.

Dechter, R.; Meiri, I.; and Pearl, J. 1991. Temporal constraint networks. *Artificial Intelligence* 49:61–95.

Fang, C.; Yu, P.; and Williams, B. C. 2014. Chance-constrained probabilistic simple temporal problems. In *Proc. 28th AAAI Conference on Artificial Intelligence*, 2264–2270.

Moffitt, M. D. 2011. On the modelling and optimization of preferences in constraint-based temporal reasoning. *Artificial Intelligence* 175(78):1390 – 1409. Representing, Processing, and Learning Preferences: Theoretical and Practical Challenges.

Morris, P.; Muscettola, N.; and Vidal, T. 2001. Dynamic control of plans with temporal uncertainty. In *Proc. 17th International Conference on Artificial Intelligence (IJCAI)*, 494–499.

Morris, P. 2006. A structural characterization of temporal dynamic controllability. In *Proc. 12th International Conference on Principles and Practice of Constraint Programming (CP)*, 375–389.

Policella, N.; Smith, S.; Cesta, A.; and Oddi, A. 2004. Generating robust schedules through temporal flexibility. In *Proc. 14th International Conference on Automated Planning & Scheduling (ICAPS)*, 209–218.

Policella, N.; Cesta, A.; Oddi, A.; and Smith, S. 2007. From precedence constraint posting to partial order schedules. *AI Communications* 20(3):163–180.

Smith, D. E.; Frank, J.; and Jonsson, A. K. 2000. Bridging the gap between planning and scheduling. *KNOWLEDGE ENGINEERING REVIEW* 15(1):2000.

Tsamardinos, I., and Pollack, M. E. 2003. Efficient solution techniques for disjunctive temporal reasoning problems. *Artificial Intelligence* 151(12):43 – 89.

Tsamardinos, I. 2002. A probabilistic approach to robust execution of temporal plans with uncertainty. In *Methods and Applications of Artificial Intelligence (Proc. 2nd Hellenic Conference on AI)*, 97–108.

Vidal, T., and Fargier, H. 1999. Handling contingency in temporal constraint networks: From consistency to controllabilities. *Journal of Experimental and Theoretical AI* 11(1):23–45.

Wah, B. W., and Xin, D. 2004. Optimization of bounds in temporal flexible planning with dynamic controllability. In *Proc. 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 40–48.

Wilson, M.; Klos, T.; Witteveen, C.; and Huisman, B. 2014. Flexibility and decoupling in simple temporal networks. *Artificial Intelligence* 214:26–44.