Contingent Durations in Temporal CSPs: from Consistency to Controllabilities

Thierry VIDAL*
Dept of Computer and Information Science (IDA)
Linköpings Universitet
S-581 83 Linköping, Sweden
e-mail: thivi@ida.liu.se

Hélène FARGIER
IRIT
Université Paul Sabatier
118, rte de Narbonne
F-31062 Toulouse, France
e-mail: fargier@irit.fr

Abstract

Temporal Constraint Networks (TCSP) allow to express minimal and maximal durations between time-points. Though being used in many research areas, this model disregards the contingent *nature of some constraints, whose effective duration cannot be decided by the system but is provided by the external world. We propose an extension of TCSP in which the classical network consistency property must be redefined in terms of* controllability*: intuitively, we would like to say that a network is controllable iff it is consistent in any* situation *(i.e. any assignment of the whole set of contingent intervals) that may arise in the external world. Three levels of controllability must be distinguished, namely the* Strong, *the* Weak *and the* Dynamic *ones. This preliminary report mainly stresses the representation and concept issues, discussing their relevance in dynamic application domains, and partially tackles the reasoning issues (complexity, algorithms and tractable subclasses).*

1 Background and overview

A large number of research areas in Artificial Intelligence are faced with the necessity of handling time in an explicit and highly expressive manner. Apart from studies dealing with logics of time, researches have been carried out on temporal algebras, i.e. formalisms solely capturing the time entities (points or intervals) and the relations among them. Constraint Satisfaction Problems [12] model them in terms of variables and constraints between them, and the main reasoning issue, on which this paper will focus, is then to check the *consistency* of the whole (i.e. check that one can find a complete assignment of the variables, called a *solution*, that satisfies the constraints). We should distinguish here between *symbolic* constraint algebras [1, 19], models dealing with *numerical* constraints [4], and some advanced proposals combining both [16, 11, 7].

We will focus in this paper on numerical temporal constraints, relying on the *Temporal CSP (TCSP)* graph-based formalism [4] that expresses them as intervals of possible values. A *duration* will be used in the following to designate one such value "assigned" [16] to a constraint (within its bounds), whereas a *date* will be classically a value assigned to a variable. This model has proven to be useful in such domains as scheduling [8], supervision [6], diagnosis and temporal databases [3], multimedia authoring environments [13], or planning. In this latter domain, the incremental planner IxTet [11] used the polynomial restriction of TCSP (namely STP) to check the temporal consistency of the plan (i.e. a partially ordered set of tasks allowing an agent to reach a given goal). The next step [18] was to take into account the inherent uncertain nature of durations of some tasks in realistic applications, distinguishing between *contingent* constraints (whose effective duration will only be observed at execution time. e.g. the duration of a task) and *free* ones (which instanciation is controlled by the agent, e.g. a delay between starting times of tasks). This will be recalled in Section 2.

Then the classical concept of consistency has to be redefined in terms of *controllability*: intuitively a network is controllable if it is consistent in the classical

*Supported by the Excellence Center for Computer Science and Systems Engineering in Linköping (ECSEL)

0-8186-7937-9/97 $10.00

sense in any *situation* that may arise in the external world, i.e. however the "nature" will decide to assign the contingent intervals. Inspired by the thorough study carried out in the framework of discrete CSPs [9] on the same kind of distinction, we exhibited two distinct properties, namely the *Strong controllability* (i.e. there exists at least one solution, that will satisfy the whole set of constraints in any situation) and the *Weak controllability* (i.e. in any situation, there exists at least one solution that satisfies the constraints in that context). But these notions fail unfortunately to encompass the reactive nature of the solution building process in dynamic domains like planning, which Section 3 will explain and argue, eventually issuing the thorough definition (which is the main contribution of this article) of a third level of controllability, the *Dynamic* one. Section 4 will only partially tackle the on-going work on theoretical complexity and practical algorithms, extended in Section 5 by a first attempt to design tractable subclasses.

2 Representation issues

2.1 The Temporal CSP and its STP restriction

In the time-point continuous algebra [19], time is represented through a set of time-points related by a number of relations, that can be be represented through a graph where nodes are time-points and edges correspond to precedence ($\preceq$) relations [10].

One can also use time-point graphs to represent numerical constraints, thanks to the TCSP formalism [4]. Here continuous binary constraints define the possible durations between two time-points by means of temporal intervals. The *STP* (Simple Temporal Problem) restriction of general TCSPs applies when those are only non-disjunctive intervals: $c_i = [l_i, u_i]$ between x and y expresses that the values of x and y must be such that $(y\text{-}x) \in [l_i, u_i]$. A TCSP is said to be consistent if one can CHOOSE for each time point a value such that all the constraints are satisfied, the resulting instanciation being one *solution* of the TCSP. Consistency checking of a general TCSP is NP-complete, but for some restrictions such as the STP, polynomial-time polynomial (e.g. the PC-2 3-consistency checking algorithm [14]) are complete.

Should the reader be interested in the overall temporal structure developped for the IxTeT planner, we suggest to have a look at [11].

2.2 The STPU model

The TCSP and STP models suit well the cases in which effective durations are under the control of the agent. If not, the problem has to be redefined in the following way [18].

Definition 2.1 (Types of constraints/variables)
A free constraint c_i *(referred as* Free *in the following) is a numerical constraint of type* $(x\text{-}y) \in [l_i, u_i]$ *that will be instanciated* by the agent *in the domain* $[l_i, u_i]$.

A contingent constraint g_i *(referred as* Ctg *in the following) is a numerical constraint of type* $(e_i - b_i) \in [l_i, u_i]$ *that will be instanciated* by the external world *in the domain* $[l_i, u_i]$.

The activated *time-points* b_i *are those which date is assigned by the agent.*

The received *time-points* e_i *are those which unpredictable date is assigned by the external world.*

Hence the domain of a *Free* can be reduced by propagation (removing values that are inconsistent with other contraints [4]) while the domain of a *Ctg* should NOT (as it would remove possibly occurring values). Anyway, considering restriction of contingent domains as a failure (as in [5], where similar distinctions are made with respect to Allen's algebra [1]) is not sufficient as a satisfiability checking process, as it will be illustrated in next section.

The previous distinctions allow us to define a new model called *Simple Temporal Problem under Uncertainty (STPU).*

Definition 2.2 (STPU)
$\mathcal{N} = (V_b, V_e, R_g, R_c)$ *represents a* STPU *with*

$V_b = \{b_1, \ldots, b_B\}$*: set of the* B *activated time-points,*

$V_e = \{e_1, \ldots, e_E\}$*: set of the* E *received time-points,*

$R_c = \{c_1, \ldots, c_C\}$*: set of the* C *Frees,*

$R_g = \{g_1, \ldots, g_G\}$*: set of the* G *Ctgs, whith* $\forall g_i = [l_i, u_i], l_i > 0$.

So a Ctg encompasses the notion of an activity (e.g. a task) which (non-null) duration is contingent. One can notice that those definitions entail the basic property that a Ctg always relates an *activated* time-point ("begin") to a *received* one ("end"), and that hence two activated [resp. received] time-points will always be related by a Free.

Please note that in the following we will use the terms *decision* to refer to the effective date of some activated time-point b_i, and *observation* for the effective duration ω_i (received by the system at time-point e_i) of the contingent interval between b_i and e_i.

3 Consistency revisited: 3 levels of controllability

Notice that in order to ease the formulation, we adopt in the following the usual misuse of merging variables and values into one unique notation.

3.1 Preliminary definitions

Definition 3.1 (Complete/partial assignments)
A control sequence *δ of the STPU is an assignment of the sole activated time-points: $\delta = \{b_1, \ldots, b_B\}$.*
A current-solution *at time-point i is a partial control sequence*
$\delta_{\prec i} = \{b_1, ..., b_{B'}\}, B' < B$, such that $\forall j = 1 \ldots B'$, $b_j \prec i$.

Hence, in planning for instance, where the solution is built reactively, a *current-solution* defines that part of *a control sequence* (i.e. a starting time for each task) currently developped until point i. This entails that $\forall \delta$ and $\forall i = 1 \ldots B$, $\exists! \delta_{\prec i} \subseteq \delta$.

Definition 3.2 (Situation)
Given that $\forall i = 1 \ldots G$, $g_i = [l_i, u_i]$,
$\Omega = [l_1, u_1] \times \ldots \times [l_G, u_G]$ will be called the space of situations, *and*
$\omega = \{\omega_1 \in [l_1, u_1], \ldots, \omega_G \in [l_G, u_G]\} \in \Omega$ will be called a situation *of the STPU.*

In other words, a *situation* represents one possible assignment of the whole set of contingent constraints. As we will argue further on, our problem can be interpreted as a "game against the nature", in which case a situation corresponds to one possible strategy carried out by the nature.

Definition 3.3 (Projection) *$\forall \omega \in \Omega$, $\mathcal{N}_\omega$ is called the* projection *of the STPU $\mathcal{N}$ in the situation ω, and is constructed by replacing every* Ctg *$g_i = [l_i, u_i]$ in $\mathcal{N}$ by the singleton $g_i = \{\omega_i\}$, with $\omega_i \in \omega$.*

Property 3.1 *A* projection *$\mathcal{N}_\omega$ defines a classical STP* [18].

The proof is immediate as a constraint in a *projection* is either a *Free* (i.e. the classical form of a STP constraint) or a singleton.

Definition 3.4 (Partial situations) *In the same way as we have defined a current-solution, we can define*
the past-situation *at point i: $\omega_{\prec i} \in \Omega_{\prec i} \subseteq \Omega$ is the set of observations prior to i.*
the situation-to-come *at point i: $\omega_{\succ i} \in \Omega_{\succ i} \subseteq \Omega$ is the set of observations subsequent to i.*

3.2 The Strong controllability

Definition 3.5 (Strong controllability)
$\mathcal{N}$ is Strongly controllable *iff*
$\exists$ a control sequence $\delta = \{b_1, ..., b_B\}$ such that $\forall \omega \in \Omega$, δ is a solution of $\mathcal{N}_\omega$.

This first definition of STPU properties tells that the STPU is *Strongly controllable* iff there exists at least one "universal" solution that fits any situation (and that hence, as far as dynamic applications are concerned, might be computed off-line beforehands). We will illustrate this property and the next ones through some small toy examples that may arise in everyday life common-sense planning.

Examples

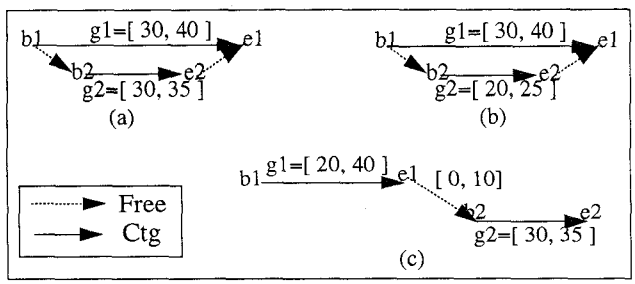

Figure 1: Strong controllability in practice

The two first drawings in Figure 1 show a *during*-like relation [1] between two Ctgs g_1 and g_2. Imagine for instance that you need some food to go and purchase downtown (task g_2), knowing that the supermarkets will be closing in 30 up to 40 minutes (contingent interval g_1). Going shopping in supermarket a will take you between 30 and 35 minutes, but if you go to supermarket b it will in any case not take you more than 25 minutes. Then it should not be so hard for you to deduce that going to supermarket b is the only way to be sure to make it on time. In other words, the first case (a) is not Strongly controllable (even though no contingent domain has been restricted, see previous section), since there exists at least one situation $\{\omega_1{=}30, \omega_2{=}35\}$ that violates the *during* relation. The second case (b) is Strongly controllable since deciding for instance to have $b_2 = b_1$ (i.e leaving now) leads to a valid solution whatever values are taken by ω_1 and ω_2.

The third case in Figure 1(c) exhibits a simple *before*-like relation [1] between two Ctgs in which the delay between the reception of e_1 and the activation of b_2 is constrained. Consider for instance that g_1 is the task Cooking and g_2 is the task Having-dinner, and you don't want to eat your dinner cold. The example

would be rejected by a *Strong controllability* checking algorithm, since one cannot fix beforehands one unique date for b_2 that will be valid in any situation: the instanciation of b_2 depends on ω_1. Which means that you cannot decide in advance at what precise time you will sit down to table, this depends on the time you'll need to cook: if it takes you only 20 minutes, then you'll necessarily start the dinner within 30 minutes after b_1. But if g_1 takes 40 minutes, then dinner will only begin at least 40 minutes after b_1. But one should notice that this example looks controllable from a planning point of view since one will have no problems deciding when to activate the second task g_2 once the first one g_1 is achieved. Hence, Strong controllability appears to be a too demanding property, calling for a "weaker" one ...

Anyway, Strong controllabilty may be relevant in specific applications where the situation is not observeable at all or where the complete control sequence must be known beforehands (e.g. when other activities processed by other agents depend on it, which for instance may arise in the production planning area).

3.3 The Weak controllability

Definition 3.6 (Weak controllability)
$\mathcal{N}$ *is* Weakly controllable *iff*
$\forall \omega \in \Omega$, $\exists \delta = \{b_1, \ldots, b_B\}$ *such that* δ *is a solution of* $\mathcal{N}_\omega$.

In other words, the STPU will remain *Weakly controllable* as far as in any given situation, there exists at least one solution (which holds in the case of Figure 1(c)). Hence, as soon as one knows the situation, one can pick out and apply the control sequence that matches the situation. But this property proves to be not so useful in planning, as we will state it hereafter.

Example

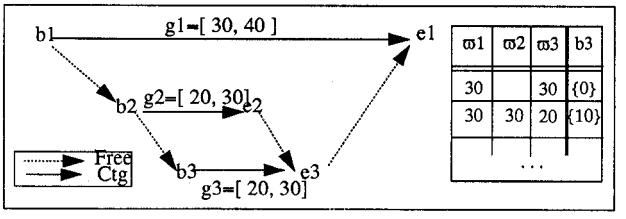

Figure 2: Drawbacks of the Weak controllability

Figure 2 exhibits a possible 3-Ctgs network. Consider for instance that g_1 represents the waiting time until your guests arrive, and g_3 your cooking activity, that should be finished before the bell is ringing. Then you realize that some ingredient is missing that should be added before the end of the dish mix, hence you must send somebody to buy it at the supermarket (task g_2). All durations of the three tasks being contingent, the question is: can you be sure to make your dish correctly and on time ?

This example can in fact be shown to be Weakly controllable (this will not be proven here). But let's have a closer look at it. If we assume for instance that $b_1 = b_2 = 0$ (which is in any case an optimal choice), then the sequence to find is reduced to the simple decision of instantiating b_3 (i.e.: when should you start cooking once you have sent somebody go shopping ?) The table next to the figure exhibits two distinct kinds of situation. The first one is when g_1 and g_3 both lasts 30 time units (no matter what value takes g_2). There is one solution in those cases: to satisfy the during-like relation between them, one is compelled to assign the value 0, and only it, to b_3 (i.e. you must start cooking now). We let the reader look into the second situation to reach the same kind of conclusion, namely in that case that the sole value 10 can be assigned to b_3 (i.e. you must wait exactly 10 minutes before starting cooking). Comparing the two situations, one can see that the decision to be taken depends upon the observation of ω_3 (i.e. on the duration of the dish making), which is necessarily unknown when b_3 is activated. Therefore in dynamic domains we would like to conclude that the network is "not controllable" (i.e. you have to choose between making sure of making a good dish, or of making it on time, but you cannot be certain of meeting both requirements ...)

Hence the property that we seek for our planning application is not Weak controllability either. But again, this property may be relevant in specific applications where the situation will be known JUST BEFORE the execution starts (consider for instance delivering delays in production planning), but one wants to know in advance that there will always be at least one feasible solution.

3.4 The Dynamic controllability

As illustrated in last paragraph, in dynamic application domains such as planning, the solution is built in the process of time, as far as one "observes" the situation at hand. Which means that decisions are to be taken whereas the situation remains partially unknown. Figure 3 illustrates it, stating that at each point in time the situation splits into the *past-situation* that is known (represented by $\omega_{\prec i}$) and the *situation-to-come* that remains unknown (and is still to be picked out from a partial space of possible situations-to-come $\Omega_{\succ i} \subseteq \Omega$). This leads to the definition of a third level of controllability.

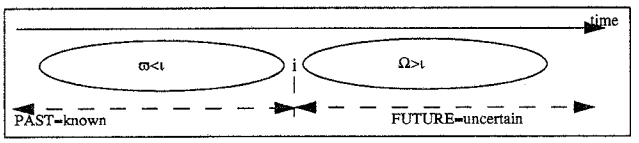

Figure 3: Dynamic observation of the situation

Definition 3.7 (Dynamic controllability)
$\mathcal{N}$ is Dynamically controllable *iff*
$\forall \omega \in \Omega$, $\exists \delta = \{b_1, ..., b_B\}$ *such that*
$\forall i \in V_b$ *with past-situation* $\omega_{\prec i} \subset \omega$, $\forall$ *possible situation-to-come* $\omega'_{\succ i} \in \Omega_{\succ i}$, $\exists$ δ' *such that the* current-solution $\delta_{\prec i} \subseteq \delta'$ *and* δ' *is a solution of* $\mathcal{N}_{\omega_{\prec i} \cup \omega'_{\succ i}}$.

In other words, each successive decision is ensured to extend to a global solution whatever occurrences remaining to be observed. Planning is not the only domain in which this applies: in multimedia authoring environments research [13], one needs make the same distinction betwen contingent and free constraints to model the temporal structure and dynamic presentation of a document. It appeared recently that our *Dynamic controllability* was the one to be considered in this framework.

3.5 The basic property relating the three levels

Property 3.2 (Implication rule)
Strong controllability $\Rightarrow$ *Dynamic controllability* $\Rightarrow$ *Weak controllability.*

Sketch of proof: The first implication is straightforward: if there exists a "universal" control sequence δ, then of course in any situation, we can choose to develop this one, and at any point in time, the current solution extends to δ itself in any possible situation-to-come.
The second implication is even more trivial: for any situation ω, if a current-solution can be extended to any situation-to-come, then it will be the case in the situation ω itself. □

In other words, when trying to satisfy the Dynamic controllability requirement, checking the Strong one will be a complete but not sound process, whereas Weak controllability checking will issue sound but uncomplete answers.

4 Complexity and algorithmic issues

This section will not be deeply developped (and some proofs will not be provided) in this preliminary report, because of the limited length and also because it refers to already published or on-going work.

4.1 Checking the Strong controllability

For an in-depth analysis of the Strong controllability checking, we invite the reader to refer back to the *Decision Graph* method described in [18]. We will only recall hereafter the basic tractability property.

Property 4.1 (Complexity of the Strong)
Checking the Strong controllability is polynomial.

Sketch of proof: The problem of deciding Strong controllability of a STPU can be represented by means of a classical STP such that the STPU is Strongly controllable iff the STP is consistent in the classical sense. The idea is to consider the relationships between tasks in the worst case, assuming that a contingent duration $d_i \in [l_i, u_i]$ is equal to u_i in any constraint of the form $x - y > d_i$ and equal to l_i in any constraint of the form $x - y < d_i$. Hence, since determining whether a STP is consistent or not is a polynomial problem, so it is for deciding Strong controllability. □

4.2 Checking the Weak controllability

Conjecture 4.1 (Complexity of the Weak)
Checking the Weak controllability is Co-NP-complete.

Sketch of proof: The Co-problem of checking Weak controllability is: is there a situation $\omega \in \Omega$ such that $\mathcal{N}_\omega$ is an inconsistent STP ? Since checking that a STP is inconsistent is a polynomial problem, this co-problem belongs to NP. Hence, Weak controllability belongs to Co-NP. The difficulty of the problem (Co-NP-*complete*) remains to be proven. □

Property 4.2 *if* $\mathcal{N}_{\{\omega_1,...,l_i,...,\omega_G\}}$ *and* $\mathcal{N}_{\{\omega_1,...,u_i,...,\omega_G\}}$ *are consistent STPs, then, for any* $v_i \in [l_i, u_i]$, $\mathcal{N}_{\{\omega_1,...,v_i,...,\omega_G\}}$ *is a consistent STP.*

Property 4.3 (Weak controllability on bounds)
A STPU is Weakly controllable iff for any $\omega^{bnd} \in \{l_1, u_1\} \times ... \times \{l_G, u_G\}$, $\mathcal{N}_{\omega^{bnd}}$ *is a consistent STP.*

If Weak controllabilty proves to be co-NP complete, one can now imagine an enumerative algorithm which checks the consistency of the projection $\mathcal{N}_{\omega^{bnd}}$ for every $\omega^{bnd} \in \{l_1, u_1\} \times ... \times \{l_G, u_G\}$. It can be processed recursively:

Sketch of algorithm: Consider $\mathcal{N}$ as a classical STP, and propagate the constraints by 3-consistency: if a Ctg $g_i = [l_i, u_i]$ is reduced, it means that at least one projection $\mathcal{N}_{\omega^{bnd}}$ is inconsistent and therefore the STPU is not Weakly controllable. Otherwise, choose one g_i and work recursively on the two simplified STPUs obtained for $g_i = \{li\}$ and for $g_i = \{u_i\}$ (notice that in a look-ahead approach one can propagate the constraints after each instanciation). The resulting complexity of this (probably non-optimal) algorithm is in $O(2^G)$.

4.3 Checking the Dynamic controllability

Characterizing the complexity of Dynamic controllability checking is a non-trivial task, that is still under process. Anyway, as a first step, it might be interesting to prove at least that this problem is not polynomial, by merely proving that the Dynamic property is more difficult to establish than the Weak one.

Then, assuming the hardness of the problem, there is a need to look for some practical method to deal with it. If one requires a complete algorithm, then it will behave exponentially in the worst case, which is undesirable in dynamic applications. But then two research tracks are worth investigating. The first one consists, in planning for instance, in using such a complete algorithm only during the execution, and only in the short-range (to keep the complexity within bounds), in a reactive manner. This of course should be combined with uncomplete methods capable of maintaining an evaluation of the feasibility of the whole plan in the long-range (before and during the execution). The second track consists in characterizing which parameters contribute to the complexity of the method in order to exhibit restricted frameworks and/or applications in which such parameters are strictly confined under some constant threshold.

This is this second track that we have chosen to investigate first. Inspired by the real-time controllers community [15], we have first designed an exponential algorithm based on the transformation of the STPU into an equivalent timed automaton encompassing all the possible execution paths. Then, considering the problem as a continuous "game against the nature", it is possible to search in the automaton, producing conditions on decision variables that must be enforced to be sure to always reach the "winning" state. If those conditions are satisfiable, then the original problem is Dynamically controllable. Characterizing the algorithmic complexity allows to extract the primary graph parameters that influence the hardness of this method. Interestingly enough those parameters appear to remain constant if one considers the already mentioned multimedia application area [13], for which one can hence expect to obtain efficient algorithms in practice.

This part of the work, only sketched hereabove, is more widely described in a technical report not yet published at the time this paper is being written. Should the reader be interested in those aspects, we suggest to contact the authors.

5 Tractable equivalence classes

Assuming that Weak and Dynamic controllabilities are more than polynomial, and that the conjecture NP $\neq$ P is true, then it is always useful to track maximal restricted representations in which the problem becomes polynomial again [17, 2, 7]. In our framework, it amounts to find the maximal equivalence classes between Weak [resp. Dynamic] and Strong controllabilities.

In [18], we have defined an algebra of the possible symbolic relations linking two Ctgs (for instance in Figure 1, the *during*-like relation will be written $g_2 \mathcal{R}_8 g_1$ and the *before* one will be $g_1 \mathcal{R}_4 g_2$) that was useful not only for building our Decision Graph, but also for proposing a first equivalence class between Weak and Strong controllabilities, restricting the possible set of relations $\mathcal{R}_i$. This first attempt suffered from uncorrect approximations, that called for a deeper analysis not yet fully achieved.

5.1 Weak / Strong equivalence class

We have mainly focused on the Weak $\equiv$ Strong equivalence classes, still relying on our definition of a precedence constraint between time-points as being $\preceq$ (before or equals) [19].

Definition 5.1 (Upper / Lower Ctgs)
A Ctg $g_i = [l_i, u_i]$ *is said to be* lower *iff* $\forall \delta$ *solution of* $\mathcal{N}_{\{\omega_1,\ldots,l_i,\ldots,\omega_G\}}$, δ *is also a solution of* $\mathcal{N}_{\{\omega_1,\ldots,u_i,\ldots,\omega_G\}}$.
A Ctg $g_i = [l_i, u_i]$ *is said to be* upper *iff* $\forall \delta$ *solution of* $\mathcal{N}_{\{\omega_1,\ldots,u_i,\ldots,\omega_G\}}$, δ *is also a solution of* $\mathcal{N}_{\{\omega_1,\ldots,l_i,\ldots,\omega_G\}}$.

Recall (section 2) that a Ctg always relates an activated time-point ("begin") to a received one ("end"), and that hence two activated (resp. received) time-points will always be related by a Free. A Ctg is always of the form:

$$e_i - b_i = \omega_i \text{ with } \omega_i \in [l_i, u_i]$$

And we only have 3 kinds of Frees:

$c_k : b_j - b_i \in [l_k, u_k]$
$c_k : b_j - e_i \in [l_k, u_k]$ i.e. $b_j - b_i - \omega_i \in [l_k, u_k]$
$c_k : e_j - e_i \in [l_k, u_k]$ i.e. $b_j + \omega_j - b_i - \omega_i \in [l_k, u_k]$

A Ctg g_i is typically *lower* when any Free related to it have one of the following forms:

$c_k : b_j - b_i \in [l_k, u_k]$
$c_k : b_j - e_i \in (-\infty, u_k]$
$c_k : e_j - e_i \in (-\infty, u_k]$

And g_i is typically *upper* when any Free related to it have one of the following forms:

$c_k : b_j - b_i \in [l_k, u_k]$
$c_k : b_j - e_i \in [l_k, +\infty)$
$c_k : e_j - e_i \in [l_k, +\infty)$

This allows us to exhibit the following characterization of the polynomial classes $PWeak_{\preceq}$ entailed by an equivalence relation with Strong controllability.

Property 5.1 ($\mathbf{PWeak_{\preceq}}$ equivalence class) *If all the Ctgs g_i are* lower *[resp.* upper*], then Strong controllability is equivalent to Weak controllability.*

Sketch of proof: *Strong controllability* $\Rightarrow$ *Weak controllability* has already been proven in the general case. Conversely, suppose that the STPU is Weakly controllable. Consider the situation ω^* where $\omega_i = l_i$ for any lower g_i and $\omega_j = u_j$ for any upper g_j (for Ctgs which are both upper and lower, one can equally choose l_i or u_i). Since the STPU is Weakly controllable, there exists a solution δ of the STP $\mathcal{N}_{\omega^*}$. It is also a solution of any $\mathcal{N}_\omega$ such that $\omega \in \Omega$, by definition of upper and lower cgts. Hence, the STPU is Strongly controllable.

We finally obtain two distinct classes. Those, together with the proof that they constitute maximal subclasses, will not be reported here but is to be developped in a longer forthcoming version of this paper. One should just be aware that expressiveness is much restricted in those classes since for instance the *during* relation must be excluded in both, and in the larger one even the *before* relation has to be discarded.

On another hand, this notion of lower and upper Ctgs is useful to enhance our Weak controllability scketch of algorithm: before enumerating, one can transform the STPU into an equivalent one for Weak controllability replacing any lower *Ctg* g_i [resp. any upper *Ctg*] by the singleton $g_i = \{l_i\}$ (resp. $g_i = \{u_i\}$]. This might also be done during the enumeration, since some *Ctgs* can become upper or lower once others have been instanciated.

5.2 On-going work

We have begun to investigate the design of maximal tractable subclasses as well for the Dynamic controllability, trying to exhibit equivalence classes between it and the Strong controllability, but this part of the work is not mature enough to be reported here.

One interesting track of research that we have been starting to look upon consists in redefining the precedence relation between time-points as being now $\prec$ (strictly before) [19]. Not only this defines an algebra closer to Allen's one [1], but it seems to permit the design of more interesting classes both for the Weak and the Dynamic controllability, that one could call $\text{PWeak}_{\prec}$ and $\text{PDynamic}_{\prec}$.

6 Conclusion

This preliminary report presented extensions of the STP model to take into account contingent durations (i.e. assignments provided by the external world). Of special interest is the new consistency property called *Dynamic controllability*, that suits well those application domains in which the solution is built in a reactive manner. If the model and the concepts are now clear, it remains to complete the theoretical complexity study. Considering the assumed hardness of the problem, and efficiency being a key issue in dynamic applications, a special interest is given in our current on-going work on completing as well the thorough characterization of maximal tractable classes and the design of efficient algorithms (uncomplete or possibly complete ones in specific applications, like in [13]) for checking the Dynamic controllability.

References

[1] J.F. Allen - *Maintaining knowledge about temporal intervals*, Communications of the ACM, 26(11):509–521, 1983.

[2] C.Bessière, A.Isli & G.Ligozat - *Global consistency in interval algebra networks: tractable subclasses*, In Proc. of the 12th European Conference on A.I. (ECAI-96), pp. 3–7, Budapest (Hungary), 1996.

[3] V.Brusoni, L.Console, B.Pernici & P.Terenziani - *LaTeR: a general purpose manager of temporal information*, Methodologies for Intelligent Systems 8. Lecture Notes in Computer Science 869:255–264, Springer Verlag, 1994.

[4] R.Dechter, I.Meiri & J.Pearl - *Temporal constraint networks*, Artificial Intelligence, 49:1–95, 1991.

[5] J.Dorn - *Hybrid temporal reasoning*, In Proc. of the 11th European Conference on A.I. (ECAI-94), pp. 625–629, Amsterdam (Netherlands), 1994.

[6] C.Dousson, P.Gaborit & M.Ghallab - *Situation recognition: representation and algorithms*, In Proc. of the 13th International Joint Conference on A.I. (IJCAI-93), Chambéry (France), 1993.

[7] T.Drakengren & P.Jonsson - *Eight maximal subclasses of Allen's interval algebra with metric time*, to appear in Journal of Artificial Intelligence Research (JAIR), 1997.

[8] D.Dubois, H.Fargier & H.Prade - *The use of fuzzy constraints in job-shop scheduling*, In Proc. of IJCAI-93 Workshop on Knowledge-Based Planning, Scheduling and Control, Chambéry (France), 1993.

[9] H.Fargier, J.Lang & T.Schiex - *Mixed constraint satisfaction: a framework for decision problems under incomplete knowledge*, In Proc. of the 12th National Conference on A.I. (AAAI-96), Portland (Oregon, USA), 1996.

[10] M.Ghallab & A.Mounir-Alaoui - *Managing efficiently temporal relations through indexed spanning trees*, In Proc. of the 11th International Joint Conference on A.I. (IJCAI-89), Detroit, USA, 1989.

[11] M.Ghallab & T.Vidal - *Focusing on a sub-graph for managing efficiently numerical temporal constraints*, in Proc. of FLorida A.I. Research Symposium (FLAIRS-95), Melbourne Beach (FL, USA), 1995.

[12] V.Kumar - *Algorithms for Constraint Satisfaction Problems: a survey*, in AI Magazine, 13(1):32–44, 1992.

[13] M.Jourdan, N.Layaïda & L.Sabry-Ismail - *Time representation and management in MADEUS: an authoring environment for multimedia documents*, in Multimedia Computing and Networking 1997, M.Freeman, P.Jardetzky & H.M.Vin ed., Proc. SPIE 3020, pp. 68–79 , February 1997.

[14] A.K.Mackworth & E.C.Freuder - *The complexity of some polynomial network consistency algorithms for constraint satisfaction problems*, Artificial Intelligence, 25(1):65–74, 1985.

[15] O.Maler, A.Pnueli & J.Sifakis - *On the synthesis of discrete controllers for timed systems*, in Proc. of the 12th Symposium on Theoretical Aspects of Computer Science (STACS), München, Germany, March 1995.

[16] I.Meiri - *Combining qualitative and quantitative constraints in temporal reasoning*, In Proc. of the 9th National Conference on A.I. (AAAI-91), Anaheim (CA, USA), 1991.

[17] B.Nebel & H.J.Bürckert - *Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra*, Journal of the ACM, 42(1):43–66, 1995.

[18] T.Vidal & M.Ghallab - *Dealing with Uncertain Durations in Temporal Constraint Networks dedicated to Planning*, In Proc. of the 12th European Conference on A.I. (ECAI-96), pp. 48-52, Budapest (Hungary), 1996.

[19] M.Vilain, H.A. Kautz & P.vanBeek - *Constraint Propagation Algorithms: a Revised Report*, Readings in Qualitative Reasoning about Physical Systems, Morgan Kaufman, 1989.