Complexity of Weak, Strong and Dynamic Controllability of CNCUs

Matteo Zavatteri[1], Romeo Rizzi[1], and Tiziano Villa[1]

[1]University of Verona, Italy

[1]{matteo.zavatteri,romeo.rizzi,tiziano.villa}@univr.it

Abstract

A *Constraint Network Under Conditional Uncertainty (CNCU)* is a formalism able to model a constraint satisfaction problem (CSP) where variables and constraints are labeled by a conjunction of Boolean variables, or *booleans*, whose truth value assignments are *out of control* and only discovered upon the execution of their related *observation points* (special kind of variables). Before the execution of the CNCU starts (i.e., the *online* assignment of values to variables), we do not know completely which constraints and variables will be taken into consideration nor in which order. *Weak controllability* implies the existence of a strategy to execute a CNCU whenever the whole uncontrollable part is known before executing. *Strong controllability* is the opposite case and implies the existence of a strategy to execute a CNCU always the same way no matter how the uncontrollable part will behave. *Dynamic controllability* implies the existence of a strategy to execute a CNCU possibly differently depending on how the uncontrollable part is behaving. In this paper we prove that weak controllability is Π_2^p-complete, strong controllability is NP-complete and dynamic controllability is PSPACE-complete.

1 Constraint Networks Under Conditional Uncertainty

Constraint networks (CNs, [5]) are a framework to model *constraint satisfaction problems (CSPs)* and check the coherence of their relational constraints saying which combinations of values assigned to the variables are permitted. The main components of a constraint network are *variables*, *domains* and *constraints* and whenever all these components are under control we simply deal with a *consistency* problem asking us to find an assignment of values to all variables satisfying all constraints.

Definition 1. A *Constraint Network (CN)* is a tuple $\langle \mathcal{X}, \mathcal{V}, D, \mathcal{C} \rangle$, where $\mathcal{X} = \{X_1, \dots, X_n\}$ is a finite set of variables, $\mathcal{V} = \{v_1, \dots, v_m\}$ is a finite set of discrete values, $D \subseteq \mathcal{X} \times \mathcal{V}$ is the *domain relation* (we write $D(X) = \{v \mid (X, v) \in D\}$ to shorten the domain of X) and $\mathcal{C} = \{R_{S_1}, \dots, R_{S_k}\}$ is a finite set of relational constraints. Each R_{S_i} is defined over a *scope* of variables $S_i \subseteq \mathcal{X}$ such that if $S_i = \{X_{i_1}, \dots, X_{i_j}\}$, then $R_{S_i} \subseteq D(X_{i_1}) \times \cdots \times D(X_{i_j})$. A CN is consistent iff every variable $X \in \mathcal{X}$ can be assigned a value $v \in D(X)$ such that all constraints in $\mathcal{C}$ are satisfied. Consistency of CNs is NP-complete [5].

Fig. 1a shows an example of consistent CN, where a possible solution is $X_1 = v_1$, $X_2 = v_3$, $X_3 = v_3$ and $X_4 = v_2$. Classic CNs do not address uncontrollable components. Indeed, when some component is out of control, satisfiability is, in general, not enough, and in such a case, we deal with a *controllability* problem. To address uncontrollable conditional constraints, CNCUs were proposed in [14, 16] as an extension of CNs to handle resource allocation problems under uncertainty in the context of business process management (BPM).

Let $\mathcal{B} = \{a, b, \dots, z\}$ be a finite set of Boolean variables, a *label* $\ell = \lambda_1 \dots \lambda_n$ is any finite conjunction of literals λ_i over the booleans in $\mathcal{B}$. The *empty label* is denoted by $\boxdot$. The *label universe of* $\mathcal{B}$, denoted by $\mathcal{B}^*$, is the set of all possible (consistent) labels drawn from $\mathcal{B}$. For instance, if $\mathcal{B} = \{a, b\}$, then $\mathcal{B}^* = \{\boxdot, a, b, \neg a, \neg b, ab, a\neg b, \neg ab, \neg a \neg b\}$ (we omit the $\wedge$ connective to ease reading). Two labels $\ell_1, \ell_2 \in \mathcal{B}^*$ are *consistent* if and only if their conjunction $\ell_1\ell_2$ is satisfiable. A label ℓ_1 *entails* a label ℓ_2 (written $\ell_1 \Rightarrow \ell_2$) if and only if all literals in ℓ_2 appear in ℓ_1 too (i.e., if ℓ_1 is more *specific* than ℓ_2).

Definition 2. A *Constraint Network Under Conditional Uncertainty (CNCU)* is a tuple $\langle \mathcal{X}, \mathcal{V}, D, \mathcal{O}, \mathcal{B}, O, L, \preceq, \mathcal{C} \rangle$, where:

- $\mathcal{X}, \mathcal{V}, D$ are the same as those given for CNs in Definition 1.
- $\mathcal{O} \subseteq \mathcal{X} = \{A?, B?, \dots\}$ is a set of *observation points.*
- $\mathcal{B} = \{a, b, \dots, z\}$ is a finite set of *booleans.* $O \colon \mathcal{B} \to \mathcal{O}$ is a bijection assigning a unique observation point $A?$ to each boolean a. When $A?$ is assigned a value $v \in D(A?)$, the truth value of a is set by Nature and no longer changes.
- $L \colon \mathcal{X} \to \mathcal{B}^*$ is a mapping assigning a label ℓ to each variable X.
- $\preceq \subseteq \mathcal{X} \times \mathcal{X}$ is a precedence relation on the variables. We write $(X_1, X_2) \in \preceq$ (or $X_1 \preceq X_2$) to express that X_1 is executed *before* X_2.
- $\mathcal{C}$ is a finite set of *conditional constraints* of the form $\ell \to R_S$, where $\ell \in \mathcal{B}^*$ and R_S is a classic relational constraint.

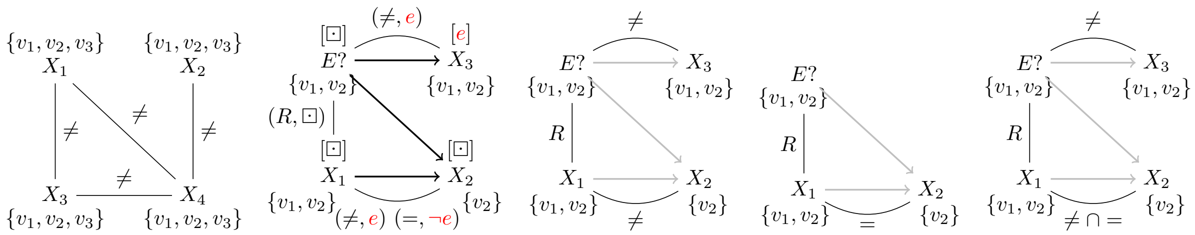
Definition 3. A CNCU is *well defined* iff all labels are consistent and:

1. For each $X \in \mathcal{X}$, if a literal a (or $\neg a$) $\in L(X)$, then $L(X) \Rightarrow L(O(a))$ and $O(a) \preceq X$.
2. For each constraint $(R_S, \ell) \in \mathcal{C}$, $\ell \Rightarrow \bigwedge_{X \in S} L(X)$ and if a literal a (or $\neg a$) $\in L(X)$, then $\ell \Rightarrow L(O(a))$.
3. For each pair $(X_1, X_2) \in \preceq$, $L(X_1) \wedge L(X_2)$ is consistent.

Regarding the notions of well-definedness (initially proposed for conditional temporal networks in [6] and then adapted to CNCUs in [14, 16]), (1) and the second part of (2) say that any label must contain the labels of the observation points associated to each proposition embedded in each contained literal (label honesty). The first part of (2) says that a label on a constraint must be at least as expressive as any label in the scope of the relation (label coherence). (3) says that we cannot impose an order between two variables not taking part together in any execution.

Fig. 1b shows the graphical representation of a well-defined CNCU specifying 4 variables $E?, X_1, X_2, X_3$, where $D(E?) = D(X_1) = D(X_3) = \{v_1, v_2\}$ and $D(X_2) = \{v_2\}$. $E?$ is an observation point whose associated boolean is e. Order edges (directed thick edges) say that $E?$ must be executed (i.e., assigned a value) before X_2 and X_3, whereas X_1 must be executed before X_2. $E?$, X_1 and X_2 are always executed as $L(E?) = L(X_1) = L(X_2) = \boxdot$ (empty label imposes no conditions). X_3 is executed if and only if e is assigned true as $L(X_3) = e$, ignored otherwise. The CNCU specifies four constraints represented as labels on constraints edges (undirected thin edges). For example, $(R, \boxdot)$ between $E?$ and X_1 (see caption) represents a relation $\boxdot \to R$ saying that if $E? = v_1$, then X_1 can be any value, whereas if $E? = v_2$, then $X_1 = v_2$. The constraint holds for any execution as its label is $\boxdot$. Instead, $(\neq, e)$ between $E?$ and X_3 says that if e is assigned true, then $E? \neq X_3$. Likewise, if e is assigned true, then $X_1 \neq X_2$, else $X_1 = X_2$.

A *scenario* $s \colon \mathcal{B} \to \{\bot, \top\}$ is a total assignment of truth values to the booleans in $\mathcal{B}$. A scenario satisfies a label ℓ (in symbols $s \models \ell$) if ℓ valuates true under the interpretation given by s. Variables and



(a) CN. (b) CNCU. (c) Proj. onto e. (d) Proj. onto $\neg e$. (e) Super proj.

Figure 1: CN, CNCU and projections. $R = \{(v_1, v_1), (v_1, v_2), (v_2, v_2)\}$. Red parts are uncontrollable.

constraints are relevant for a scenario s if their labels are satisfied by s. A *projection* of a CNCU onto a scenario s is a classic constraint network (plus the partial order between the survived variables) in which we keep only variables and constraints relevant for s. For instance, Fig. 1c and Fig. 1d show the 2 possible projections of Fig. 1b onto $s(e) = \top$ and $s(e) = \bot$. A CNCU is *weakly controllable* if any projection admits a total order on the variables and is consistent in the CN-sense. Fig. 1b is weakly controllable. If $s(e) = \top$ (Fig. 1c), then $E? = v_1$, $X_1 = v_1$, $X_2 = v_2$ and $X_3 = v_2$ (in this order). If $s(e) = \bot$ (Fig. 1d), then $E? = v_1$, $X_1 = v_2$ and $X_2 = v_2$ (again, in this order). A CNCU is *strongly controllable* if the CN *super-projection* obtained by wiping out labels on variables and constraints of the original CNCU admits a total order and is consistent in the CN sense. Fig. 1b is not strongly controllable. The super projection (Fig. 1e) contains $(\neq, e)$ and $(=, \neg e)$ between X_1 and X_2 (of the original CNCU) whose intersection yields an empty relation.

For weak and strong controllability we just make sure that a total order for $\mathcal{X}$ exists (that's why we grayed it in Fig. 1c, Fig. 1d and Fig. 1e). Instead, dynamic controllability of CNCUs is a matter of order.

A CNCU is *dynamically controllable* if its variables are executed in possible different orderings and assigned possibly different values while executing depending on the scenario being generated such that the final total order on $\mathcal{X}$ meets $\preceq$ and all relevant constraints are satisfied. Note that despite variables specify labels, assigning values to those that have become irrelevant is superfluous but not wrong (we can always ignore those assignments at the end of the execution). Fig. 1b is uncontrollable if X_1 is executed before $E?$. Indeed, if $X_1 = v_1$ and $E? = v_1$ and then $s(e) = \bot$ (or $X_1 = v_2$ and $E? = v_2$ and then $s(e) = \top$), there is not valid value for X_2 satisfying $(=, \neg e)$ or $(\neq, e)$, respectively. Instead, the CNCU is dynamically controllable if $E?$ is executed first. A possible *execution strategy* is: $E? = v_1$ (always). If $s(e) = \top$, then $X_1 = v_1$, $X_2 = v_2$ and $X_3 = v_2$, whereas if $s(e) = \bot$ then $X_1 = v_2$ and $X_2 = v_2$ (we omit irrelevant variables).

2 Complexity of Weak, Strong and Dynamic Controllability

We introduce a lemma we will rely on when proving hardness results. Such a lemma reduces in polynomial time any quantified boolean formula (QBF) to an equivalent CNCU.

Lemma 1. *Any quantified boolean formula can be reduced to a CNCU in polynomial time.*

Proof. Let $\Phi \equiv Q_1 x_1, \ldots, Q_n x_n \varphi$ be a quantified boolean formula (QBF) where $Q_i \in \{\exists, \forall\}$ $(1 \leq i \leq n)$ and $\varphi \equiv C_1 \wedge \cdots \wedge C_m$ is a 3-CNF specifying m clauses over the variables $x_1, \ldots, x_n$. We reduce Φ to a CNCU $\mathcal{N}_\Phi = \langle \mathcal{X}, \mathcal{V}, D, \mathcal{O}, \mathcal{B}, O, L, \preceq, \mathcal{C} \rangle$ as follows. For each "$\exists x$" in Φ, we add a variable X to $\mathcal{X}$ such that $D(X) = \{0, 1\}$ and $L(X) = \boxdot$. For each "$\forall y$" in Φ we add a boolean y to $\mathcal{B}$, an observation point $B_y?$ to $\mathcal{X}$ and to $\mathcal{O}$ and a variable Y to $\mathcal{X}$ such that $O(y) = B_y?$, $D(B_y?) = \{0\}$, $D(Y) = \{0, 1\}$ and $L(B_y?) = L(Y) = \boxdot$. We impose that $B_y?$ executes before Y by adding $(B_y?, Y)$ to $\preceq$. We add two conditional relational constraints $y \to R_Y^\top$ and $\neg y \to R_Y^\bot$ to $\mathcal{C}$, where $R_Y^\top = \{(1)\}$ and $R_Y^\bot = \{(0)\}$ to model that if $s(y) = 1$, then $Y = 1$ in the CNCU (meaning that $y = 1$ in Φ), whereas if $s(y) = 0$, then $Y = 0$ in the CNCU (meaning that $y = 0$ in Φ). In this way, the value assignment to Y in the CNCU is forced to "mirror" the uncontrollable truth value assignment to the boolean y. We add $n - 1$ precedence

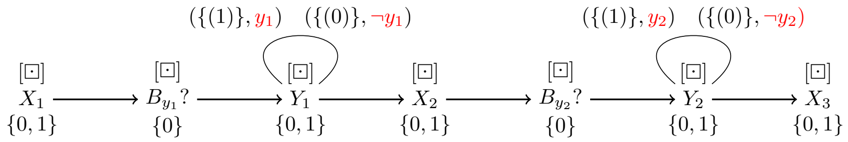


Figure 2: Example of reduction from $\Phi \equiv \exists x_1 \forall y_1 \exists x_2 \forall y_2 \exists x_3 (x_1 \vee \neg y_2 \vee \neg y_1) \wedge (x_2 \vee \neg x_3 \vee y_1)$. We encode the first clause as $\boxdot \to R_1$ and $R_1 \equiv D(X_1) \times D(Y_2) \times D(Y_1) \setminus \{(0,1,1)\}$ and the second one as $\boxdot \to R_2$ where $R_2 \equiv D(X_2) \times D(X_3) \times D(Y_1) \setminus \{(0,1,0)\}$. Being ternary, $\boxdot \to R_1$ and $\boxdot \to R_2$ are not shown.

constraints to connect each previous discussed "gadget" encoding a quantified variable of Φ to the next one according to the order in which these variables appear in the quantified part of Φ. Finally, for each clause C_i we add a relational constraint $\boxdot \to R_{S_i}$ such that the scope S_i contains the three variables embedded in the literals appearing in C_i, whereas the set of tuples is the cross product of the domains of such variables minus the unique tuple falsifying the clause (each of these relations has exactly $2^3 - 1$ tuples). Now it is easy to see that the reduction runs in polynomial time. Fig. 2 provides an example. $\square$

Theorem 1. *Weak controllability of CNCUs is Π_2^p-complete.*

Proof. **Hardness:** Lemma 1 allows us to reduce in polynomial time a QBF having the form $\forall y_1, \ldots \forall y_n, \exists x_1, \ldots \exists x_m \varphi$. Solving such an instance of QBF is known to be Π_2^p-complete. The reduction constructs a CNCU which is weakly controllable iff Φ is satisfiable. **Membership:** Any disqualification consists of a specific scenario s for which there is no solution to the corresponding projection. Once we have s, the inconsistency of the projection can be verified in non-deterministic polynomial time by using topological sort plus any decision procedure to check consistency of CNs. $\square$

Theorem 2. *Strong controllability of CNCUs is NP-complete.*

Proof. We already discusses that a CNCU is strongly controllable if it admits a total order and the CN *super-projection* is consistent. Such a projection is computable in polynomial time (Algorithm 1, line 1).

Algorithm 1: `CncuSC`($\mathcal{N}$)

Input: A CNCU $\mathcal{N} = \langle \mathcal{X}, \mathcal{V}, D, \mathcal{O}, \mathcal{B}, O, L, \preceq, \mathcal{C} \rangle$
Output: Yes, if $\mathcal{N}$ is strongly controllable. No otherwise.
1 $\mathcal{N}^* = \langle \mathcal{X}, \mathcal{V}, D, \{R_S \mid \ell \to R_S \in \mathcal{C}\} \rangle$ ▷ Compute super-projection
2 **if** *no total order on* $\mathcal{X}$ *meets* $\preceq$ **then return** No ▷ Topological Sort
3 **return** `CN-Consistency`($\mathcal{N}$) ▷ Any consistency decision procedure for CNs

We compute a total order by means of a run of Topological Sort. If none exists, the CNCU is not strongly controllable. A strong assignment for a CNCU is an assignment satisfying all constraints regardless of their labels. We claim that there exists a strong assignment for $\mathcal{N}$ iff there exists an assignment for the corresponsing super-projection $\mathcal{N}^*$. The former because a strong solution satisfying all constraints of $\mathcal{N}$ satisfies trivially their intersection in $\mathcal{N}^*$, the latter because a solution to $\mathcal{N}^*$ is a solution satisfying the intersection of all constraints of $\mathcal{N}$, therefore, a solution satisfying all constraints of $\mathcal{N}$. Therefore, strong controllability is equivalent to CN-consisteny which is NP-complete. $\square$

Theorem 3. *Dynamic controllability of CNCUs is PSPACE-complete.*

Proof. **Hardness:** Once again, Lemma 1 allows us to reduce in polynomial time a QBF without restrictions on the number of alternations of the quantifiers to a CNCU that is dynamically controllable iff Φ is satisfiable. **Membership:** Algorithm 2 is a polynomial space algorithm to decide dynamic controllability of any CNCU. In Algorithm 2, s is a scenario, α an *assignment* $\alpha \colon \mathcal{X} \to \mathcal{V}$ of values to variables, ω is a permutation $(X_{i_1}, \ldots, X_{i_n})$ of $\mathcal{X}$ (with $n = |\mathcal{X}|$). We write $\omega(X_{i_j})$ to refer to the index j of X_{i_j} in ω. Given a CNCU $\mathcal{N} = \langle \mathcal{X}, \mathcal{V}, D, \mathcal{O}, \mathcal{B}, O, L, \preceq, \mathcal{C} \rangle$, and a triple (s, α, ω) we say that $(s, \alpha, \omega) \models \mathcal{N}$ iff:

1. for each $X \in \mathcal{X}$, $\alpha(X) \in D(X)$
2. for each $\ell \to R_{\{X_{i_1}, \ldots, X_{i_n}\}} \in \mathcal{C}$, if $s \models \ell$, then $(\alpha(X_{i_1}), \ldots, \alpha(X_{i_n})) \in R_{\{X_{i_1}, \ldots, X_{i_n}\}}$

3. for each $(X, Y) \in \preceq$, $\omega(X) < \omega(X)$.

$\square$

Algorithm 2: `CncuDC`$(\mathcal{N})$

Input: A CNCU $\mathcal{N} = \langle \mathcal{X}, \mathcal{V}, D, \mathcal{O}, \mathcal{B}, O, L, \preceq, \mathcal{C} \rangle$
Output: Yes, if $\mathcal{N}$ is dynamically controllable. No otherwise.

`CncuDC` $(\mathcal{N})$
  Let s, α, ω be an empty scenario, assignment and ordering.
  return `Explore`$(\mathcal{N}, \mathcal{X}, s, \alpha, \omega)$

`Explore` $(\mathcal{N}, \mathcal{X}, s, \alpha, \omega)$
  if $\mathcal{X} = \emptyset$ **then return** $(s, \alpha, \omega) \models \mathcal{N}$ ▷ leaf check
  for $X \in \mathcal{X}$ **do** ▷ pick a variable
    $\omega' \leftarrow (\omega, X)$ ▷ append X to the current ordering
    for $v \in D(X)$ **do** ▷ look for a value to assign to X
      $\alpha' \leftarrow \alpha \cup \{\alpha'(X) \leftarrow v\}$ ▷ extend current plan
      if $X \in \mathcal{O}$ **then** ▷ case 1
        Let x be the boolean associated to X
        $s' \leftarrow s \cup \{s(x) \leftarrow \top\}$ ▷ extend scenario (positive case)
        $s'' \leftarrow s \cup \{s(x) \leftarrow \bot\}$ ▷ extend scenario (negative case)
        if `Explore`$(\mathcal{N}, \mathcal{X} \setminus \{X\}, s', \alpha', \omega') \wedge$ `Explore`$(\mathcal{N}, \mathcal{X} \setminus \{X\}, s'', \alpha', \omega')$ **then**
          return Yes
      if $X \notin \mathcal{O} \wedge$ `Explore`$(\mathcal{N}, \mathcal{X} \setminus \{X\}, s, \alpha', \omega')$ **then return** Yes ▷ case 2
  return No ▷ "no strategy" from subtree

3 Conclusions and Future Work

We discussed the computational complexity of weak, strong and dynamic controllability of CNCUs. Weak controllability is Π_2^p-complete, strong controllability is NP-complete (as it is equivalent to a satisfiability problem), whereas dynamic controllability is PSPACE-complete.

As future work we plan to compare with complexity results for other classes of (temporal)-constraint networks such as those discussed (or employed) in [1, 2, 3, 4, 7, 8, 9, 10, 11, 12, 13, 15, 14, 16, 17].

References

[1] M. Cairo, C. Combi, C. Comin, L. Hunsberger, R. Posenato, R. Rizzi, and M. Zavatteri. Incorporating decision nodes into conditional simple temporal networks. In *24th International Symposium on Temporal Representation and Reasoning, TIME 2017*, volume 90 of *LIPIcs*, pages 9:1–9:17. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[2] C. Combi, R. Posenato, L. Viganò, and M. Zavatteri. Access controlled temporal networks. In *9th International Conference on Agents and Artificial Intelligence, ICAART 2017*, pages 118–131. INSTICC, ScitePress, 2017.

[3] C. Combi, R. Posenato, L. Viganò, and M. Zavatteri. Conditional simple temporal networks with uncertainty and resources. *Journal of Artificial Intelligence Research*, 64:931–985, 2019.

[4] C. Combi, L. Viganò, and M. Zavatteri. Security constraints in temporal role-based access-controlled workflows. In *6th ACM Conference on Data and Application Security and Privacy*, CODASPY '16, pages 207–218. ACM, 2016.

[5] R. Dechter. *Constraint processing*. Elsevier, 2003.

[6] L. Hunsberger, R. Posenato, and C. Combi. A sound-and-complete propagation-based algorithm for checking the dynamic consistency of conditional simple temporal networks. In *22nd International Symposium on Temporal Representation and Reasoning, TIME 2015*, pages 4–18. IEEE CPS, 2015.

[7] M. Zavatteri. Conditional simple temporal networks with uncertainty and decisions. In *24th International Symposium on Temporal Representation and Reasoning, TIME 2017*, volume 90 of *LIPIcs*, pages 23:1–23:17. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[8] M. Zavatteri. *Temporal and Resource Controllability of Workflows Under Uncertainty.* PhD thesis, University of Verona, Italy, 2018.

[9] M. Zavatteri. Temporal and resource controllability of workflows under uncertainty. In *Proceedings of the Dissertation Award, Doctoral Consortium, and Demonstration Track at BPM 2019 co-located with 17th International Conference on Business Process Management, BPM 2019*, volume 2420 of *CEUR Workshop Proceedings*, pages 9–14. CEUR-WS.org, 2019.

[10] M. Zavatteri, C. Combi, R. Posenato, and L. Viganò. Weak, strong and dynamic controllability of access-controlled workflows under conditional uncertainty. In *Business Process Management - 15th International Conference, BPM 2017*, pages 235–251. Springer, 2017.

[11] M. Zavatteri, C. Combi, R. Rizzi, and L. Viganò. Hybrid sat-based consistency checking algorithms for simple temporal networks with decisions. In *26th International Symposium on Temporal Representation and Reasoning, TIME 2019*, volume 147, page 2:1–2:17. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019.

[12] M. Zavatteri, C. Combi, and L. Viganò. Resource controllability of workflows under conditional uncertainty. In *Business Process Management Workshops, AI4BPM 2019 (to appear)*. Springer.

[13] M. Zavatteri, R. Rizzi, and T. Villa. Strong controllability of temporal networks with decisions. In *1st Workshop on fOrmal VERification, Logic, Automata, and sYnthesis, 2019, OVERLAY 2019 (to appear)*. CEUR-WS.org, 2019.

[14] M. Zavatteri and L. Viganò. Constraint networks under conditional uncertainty. In *10th International Conference on Agents and Artificial Intelligence — Volume 2, ICAART 2018*, pages 41–52. SciTePress, 2018.

[15] M. Zavatteri and L. Viganò. Conditional simple temporal networks with uncertainty and decisions. *Theoretical Computer Science*, 797:77–101, 2019.

[16] M. Zavatteri and L. Viganò. Conditional uncertainty in constraint networks. In *Agents and Artificial Intelligence*, pages 130–160. Springer, 2019.

[17] M. Zavatteri and L. Viganò. Last man standing: Static, decremental and dynamic resiliency via controller synthesis. *Journal of Computer Security*, 27(3):343–373, 2019.