

On model-checking durational Kripke structures

F. Laroussinie*, N. Markey^o, Ph. Schnoebelen*

<http://www.lsv.ens-cachan.fr>

* [LSV](#), [ENS de Cachan](#) & [CNRS UMR 8643](#)

^o [LIFO](#), [Univ. d'Orléans](#) & [CNRS FRE 2490](#)

Verifying real-time systems $A \models \varphi$

Verifying real-time systems $A \models \varphi$

Describing real-time systems

(where quantitative information about timing is required)

Ex: Time-out

Verifying real-time systems $A \models \varphi$

Describing real-time systems

(where quantitative information about timing is required)

Ex: Time-out

Expressing quantitative properties

(over the timing of actions)

Ex: “any problem is followed by an alarm in at most 20 time units”

Timed Specification Languages

“any problem is followed by an alarm in at most 20 time units”

- Temporal logics with **subscripts**.

$$\text{AG}(\text{problem} \Rightarrow \text{AF}_{\leq 20} \text{ alarm})$$

- Temporal logics with **clocks**.

$$\text{AG}\left(\text{problem} \Rightarrow (x \text{ in } \text{AF}(x \leq 20 \wedge \text{alarm}))\right)$$

Cost of verifying timed models

	Kripke structures	Timed automata
Reachability:	NLOGSPACE-C	PSPACE-C
$(T)CTL$:	$O(S \cdot \varphi)$	PSPACE-C
$(T)LTL$:	PSPACE-C	undecidable if $T = \mathbb{R}$ or EXPSPACE-C if $T = \mathbb{N}$
AF- μ -cal.	$O(S \cdot \varphi)$	EXPTIME-C

Cost of verifying timed models

	Kripke structures	Timed automata
Reachability:	NLOGSPACE-C	PSPACE-C
$(T)CTL$:	$O(S \cdot \varphi)$	PSPACE-C
$(T)LTL$:	PSPACE-C	undecidable if $T = \mathbb{R}$ or EXPSPACE-C if $T = \mathbb{N}$
AF- μ -cal.	$O(S \cdot \varphi)$	EXPTIME-C

Using timed automata induce a complexity blowup! [[AH92](#), [AH94](#), [ACD93](#), [AL99](#)]

Cost of verifying timed models

	Kripke structures	Timed automata
Reachability:	NLOGSPACE-C	PSPACE-C
$(T)CTL$:	$O(S \cdot \varphi)$	PSPACE-C
$(T)LTL$:	PSPACE-C	undecidable if $T = \mathbb{R}$ or EXPSPACE-C if $T = \mathbb{N}$
AF- μ -cal.	$O(S \cdot \varphi)$	EXPTIME-C

Using timed automata induce a complexity blowup! [[AH92](#), [AH94](#), [ACD93](#), [AL99](#)]

Is it possible to have quantitative constraints without such a blowup?

Modelling with simpler models

It is possible to use **classical Kripke structures** as timed models.

There is **no inherent concept of time**: time elapsing is encoded by events.

Modelling with simpler models

It is possible to use **classical Kripke structures** as timed models.

There is **no inherent concept of time**: time elapsing is encoded by events.

For example:

- each transition = one time unit
- or: a “**tick**” proposition labels states where one t.u. elapses.

Modelling with simpler models

It is possible to use [classical Kripke structures](#) as timed models.

There is [no inherent concept of time](#): time elapsing is encoded by events.

For example:

- [each transition = one time unit](#)
- or: a [“tick”](#) proposition labels states where one t.u. elapses.

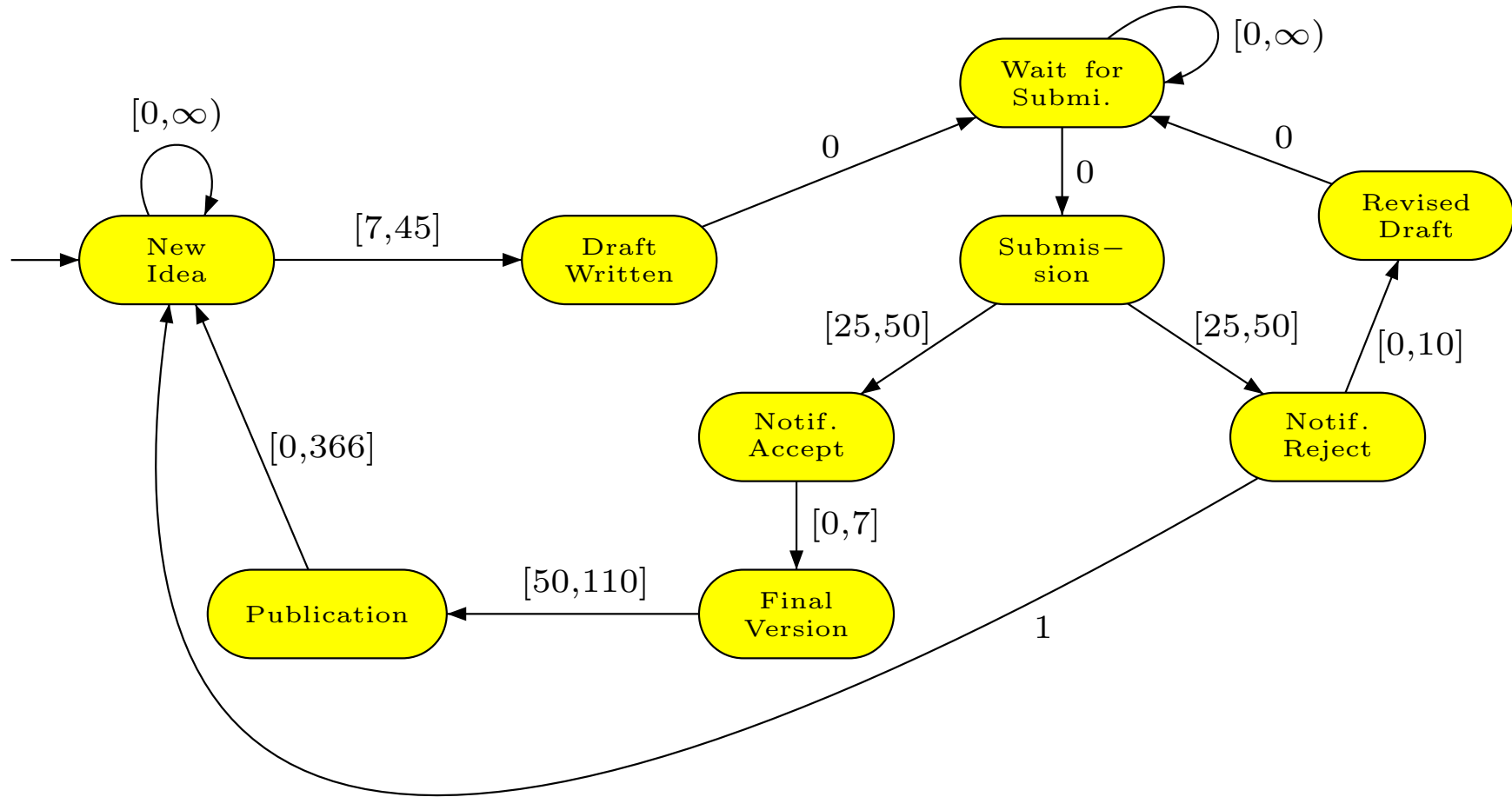
Model-checking can be polynomial-time! ([\[EMSS92, LST00\]](#))

[Is it possible to have more expressive models and temporal logics while staying polynomial-time?](#)

Outlines

- our model: **Durational Kripke Structure**
- Model-checking $TCTL$ is Δ_2^p -complete
Model-checking $TCTL_{\leq, \geq}$ is P -complete
- And $TLTL$, $TCTL^*$, $TCTL^+$...
- Conclusion

Durational Kripke Structures



Durational Kripke Structures (DKS)

A durational Kripke structure S is $\langle Q, R, l \rangle$ where

- Q is a (finite) set of *states*,
- $R \subseteq Q \times \mathcal{I} \times Q$ is a total *transition relation with duration*
- $l : Q \rightarrow 2^{AP}$ labels every state with a subset of AP .

\mathcal{I} is the set of intervals of the form “[n, m]” or “[n, ∞)”
(with $n, m \in \mathbb{N}$)

AP is the set of atomic propositions

Semantics of DKS

A transition $q \xrightarrow{[n,m]} q'$ in the model means that “moving from q to q' takes some duration d in $[n, m]$.”

The behaviour is: $q \xrightarrow{d} q'$ ($d \in \mathbb{N}$: Time is **discrete**)

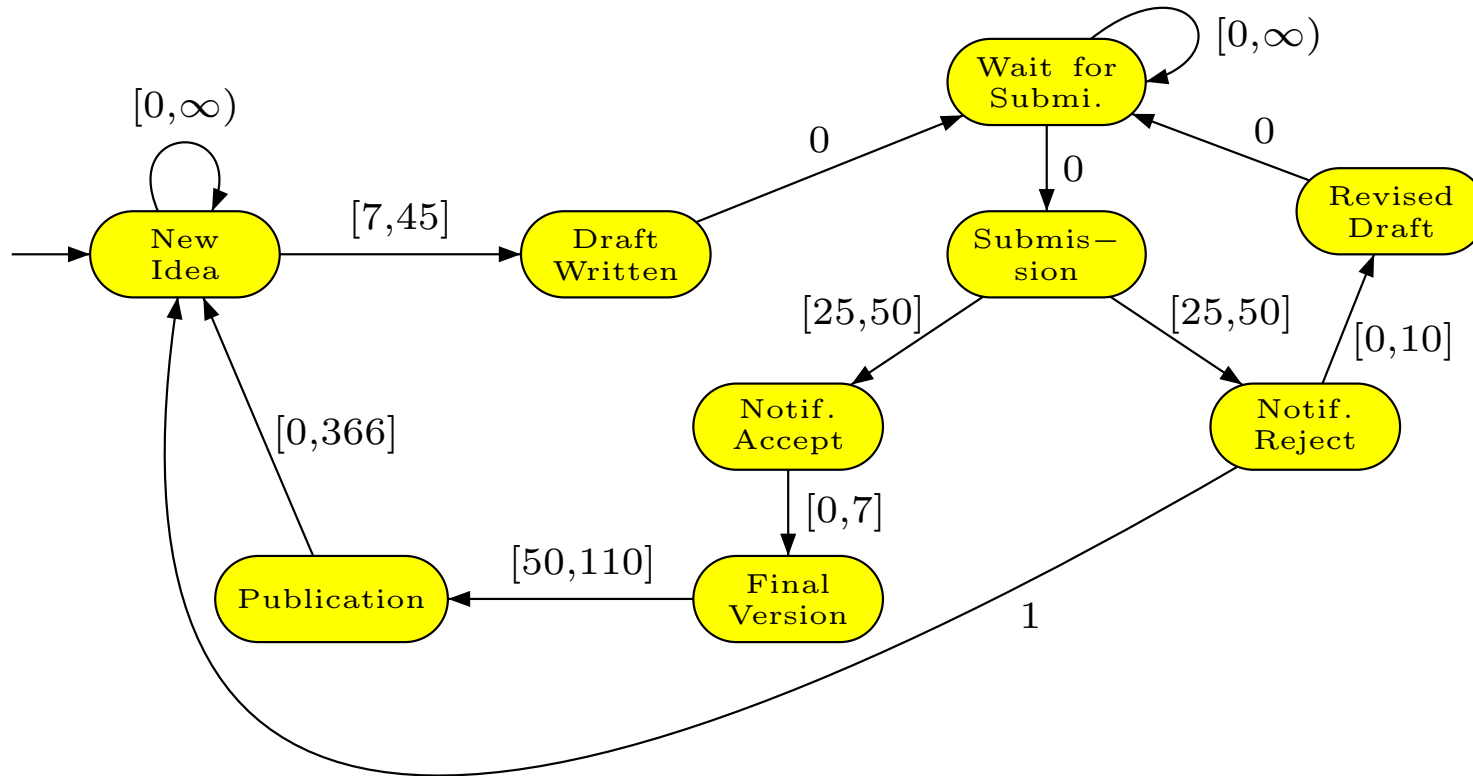
A path π in a DKS is:

$$\pi = q_0 \xrightarrow{d_0} q_1 \xrightarrow{d_1} q_2 \dots \text{ with } q_i \xrightarrow{d_i} q_{i+1} \in R \text{ for all } i.$$

The **length** of a finite path $\pi = q_0 \xrightarrow{d_0} q_1 \xrightarrow{d_1} q_2 \dots q_n$ is n .

The **duration** of π (denoted **Time**(π)) is $d_0 + \dots + d_{n-1}$.

Semantics of DKS



$\text{New_Idea} \xrightarrow{15} \text{Draft_Written} \xrightarrow{0} \text{Wait_for...} \xrightarrow{20} \text{Wait_for...} \xrightarrow{0}$
 $\text{Submission} \xrightarrow{27} \text{Notif. of acc.}$

Variants of DKS in literature

- **tight DKS** : all intervals are singletons ($q \xrightarrow{d} q$).
“state graphs” in [AH94] or “timed KS” in [ET99].
- **small-step DKS** (ssDKS): all steps have duration 0 or 1.
Similar to “KS + tick” in [LST00] and KS in [EMSS92].

Variants of DKS in literature

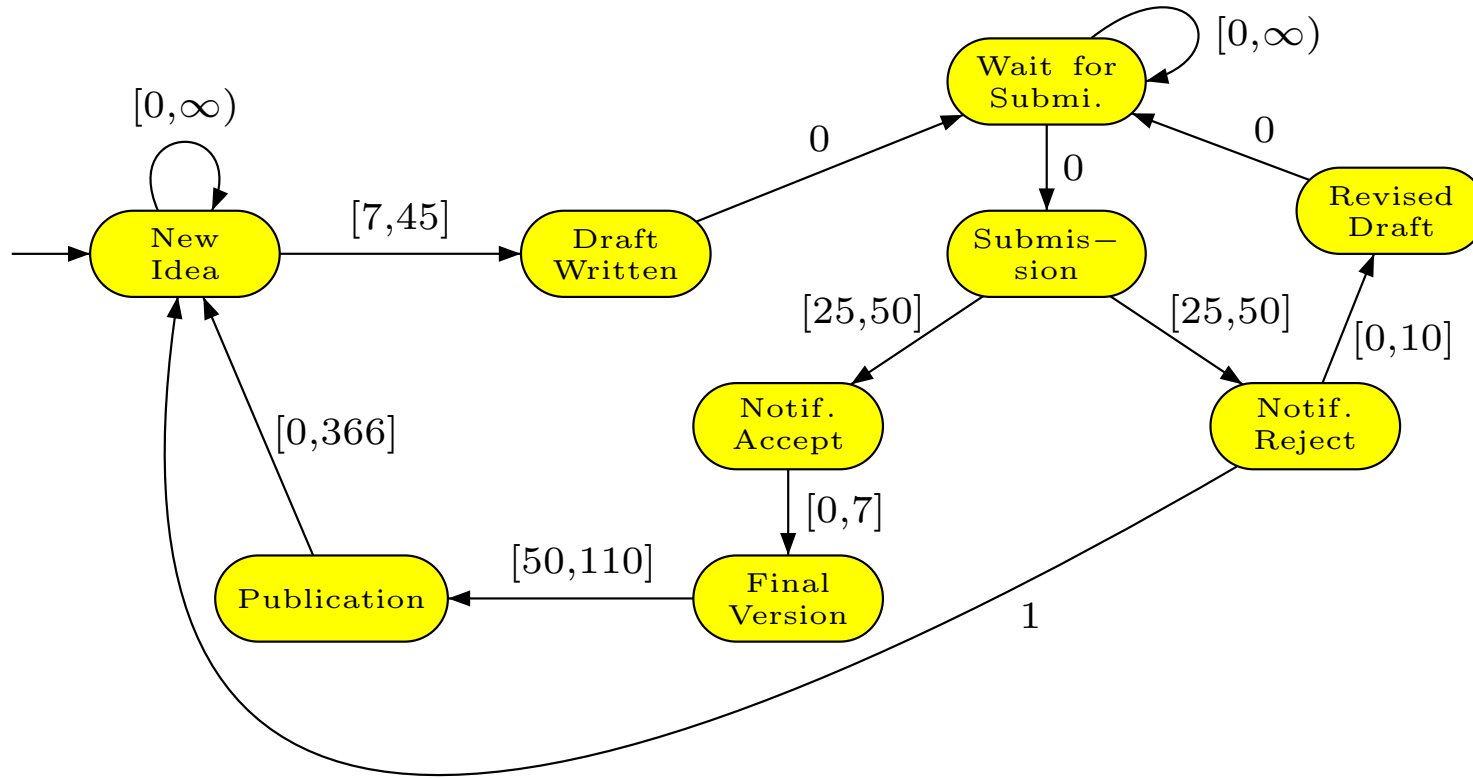
- **tight DKS** : all intervals are singletons ($q \xrightarrow{d} q$).
“state graphs” in [AH94] or “timed KS” in [ET99].
- **small-step DKS** (ssDKS): all steps have duration 0 or 1.
Similar to “KS + tick” in [LST00] and KS in [EMSS92].

We have two specific properties over the small-step DKSs:

- Durations of shortest paths are less than $|Q| - 1$,
- Time progresses smoothly along paths:
a path π of duration $d = d_1 + d_2$ can always be decomposed into $\pi' \cdot \pi''$
such that $\text{Time}(\pi') = d_1$ and $\text{Time}(\pi'') = d_2$.

These properties do not hold for DKSs !

Expressing Properties over DKS



“whenever a notification is received, either publication or submission occurs in less than 150 days ? ”

Definition of *TCTL*

TCTL formulae are built from:

- atomic proposition (For ex. **Submission**, **Notification**, **Publication**)
 - boolean combinators (\wedge , \vee , \neg)
 - **EX** operator
 - **E** - $U_{\sim c}$ - and **A** - $U_{\sim c}$ -
- + all the standard abbreviations: **AG** $_{\sim c}$, **AF** $_{\sim c}$ etc.

Definition of $TCTL$

$TCTL$ formulae are built from:

- atomic proposition (For ex. **Submission**, **Notification**, **Publication**)
- boolean combinators (\wedge , \vee , \neg)
- **EX** operator
- **E** - **U**_{~c} - and **A** - **U**_{~c} -

+ all the standard abbreviations: **AG**_{~c}, **AF**_{~c} etc.

$q \models E\varphi U_{\sim c}\psi$ iff there is a run $\pi : q = q_0 \xrightarrow{d_0} q_1 \xrightarrow{d_1} q_2 \cdots$ and an integer n s.t.

$$\text{Time}(\pi|_n) \sim c, q_n \models \psi, \text{ and } q_i \models \varphi \text{ for all } 0 \leq i < n$$

Definition of $TCTL$

$TCTL$ formulae are built from:

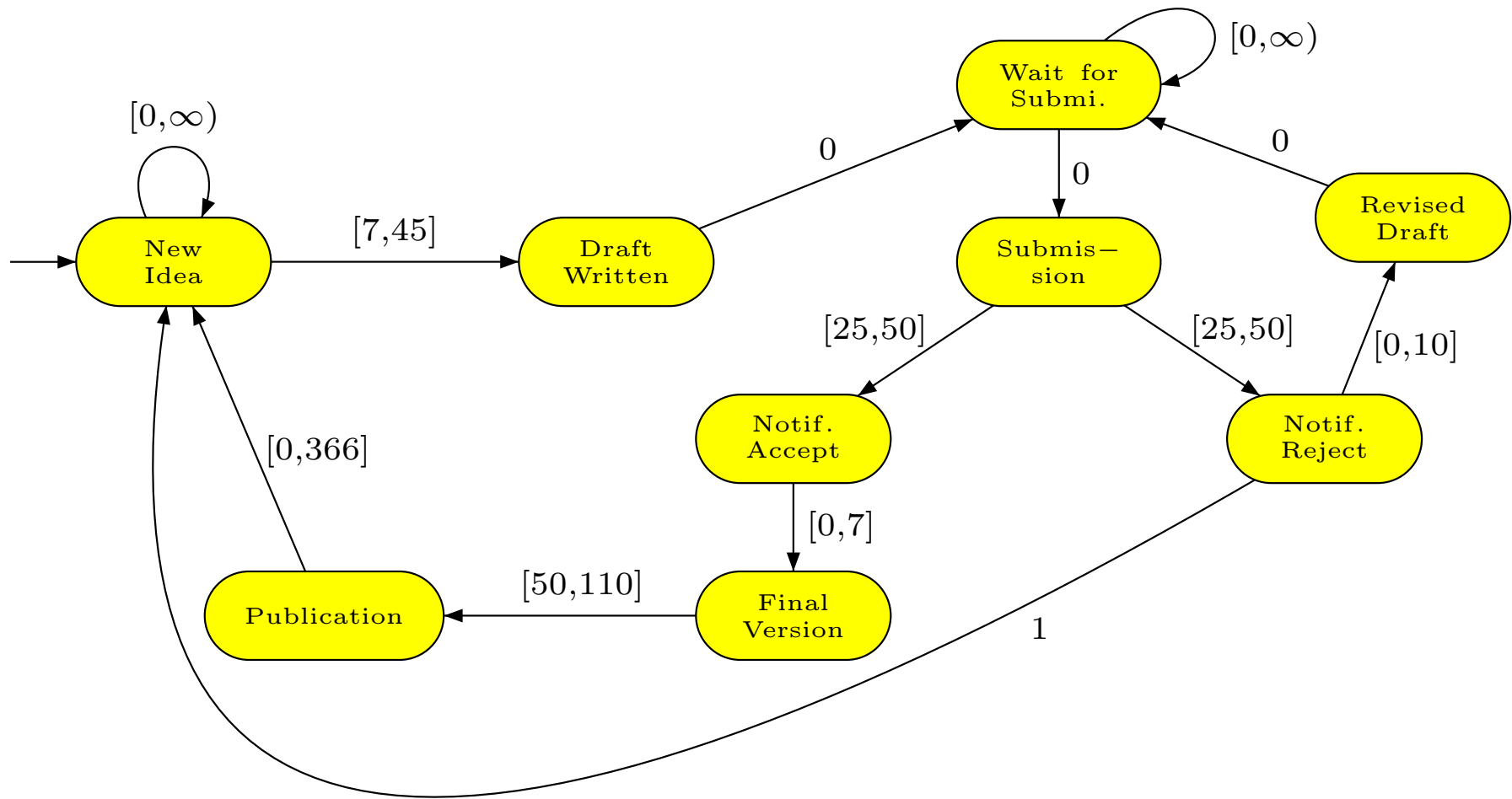
- atomic proposition (For ex. **Submission**, **Notification**, **Publication**)
- boolean combinators (\wedge , \vee , \neg)
- **EX** operator
- **E** - $U_{\sim c}$ - and **A** - $U_{\sim c}$ -

+ all the standard abbreviations: $AG_{\sim c}$, $AF_{\sim c}$ etc.

“whenever a notification is received, either publication or submission occurs in less than 150 days ? ”

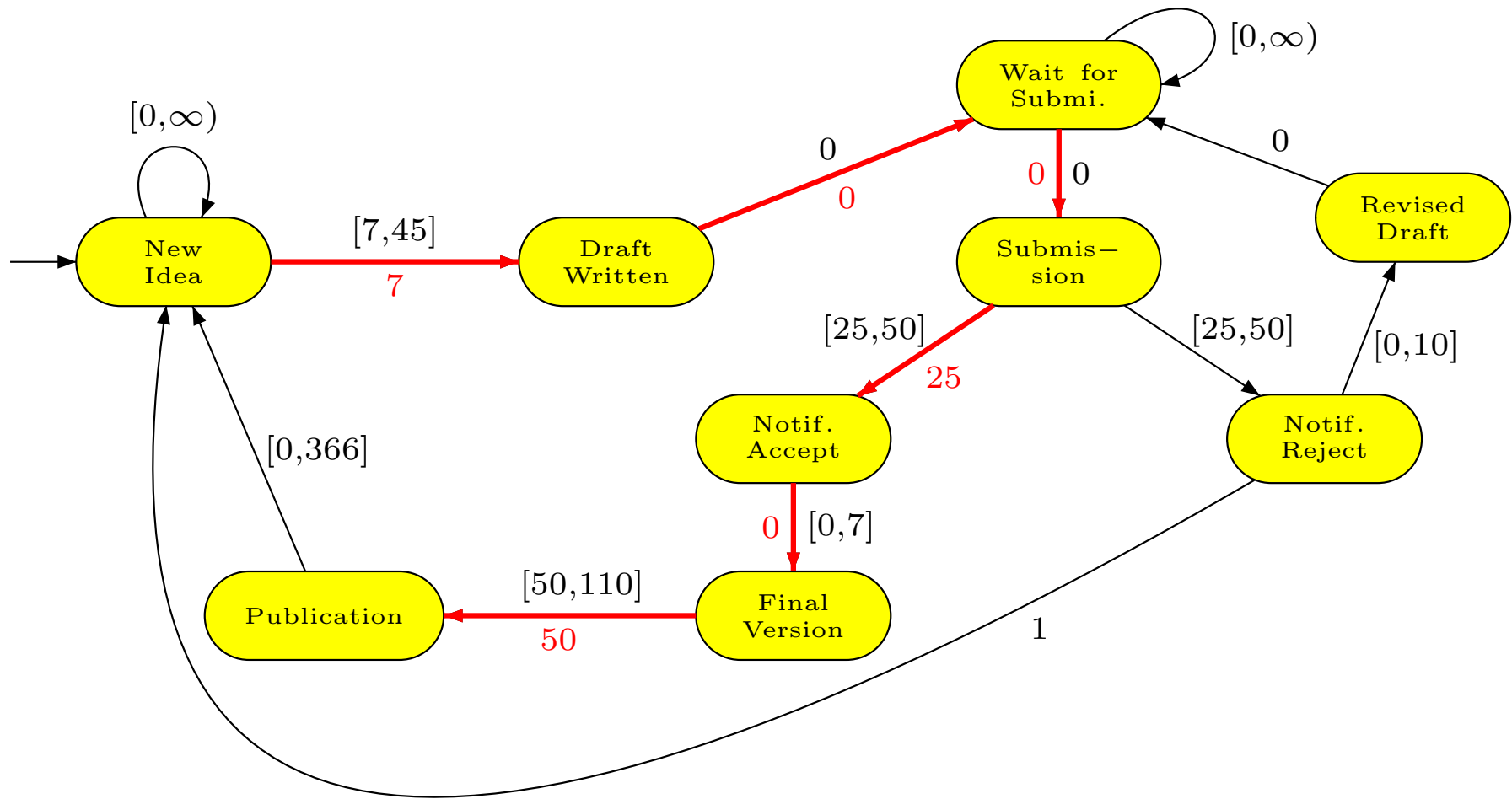
$$AG \left(\text{Notification} \Rightarrow AF_{\leq 150} (\text{Publication} \vee \text{Submission}) \right)$$

Exercise - 1



$AG(\text{New_Idea} \Rightarrow \neg EF_{<100} \text{Publication})$

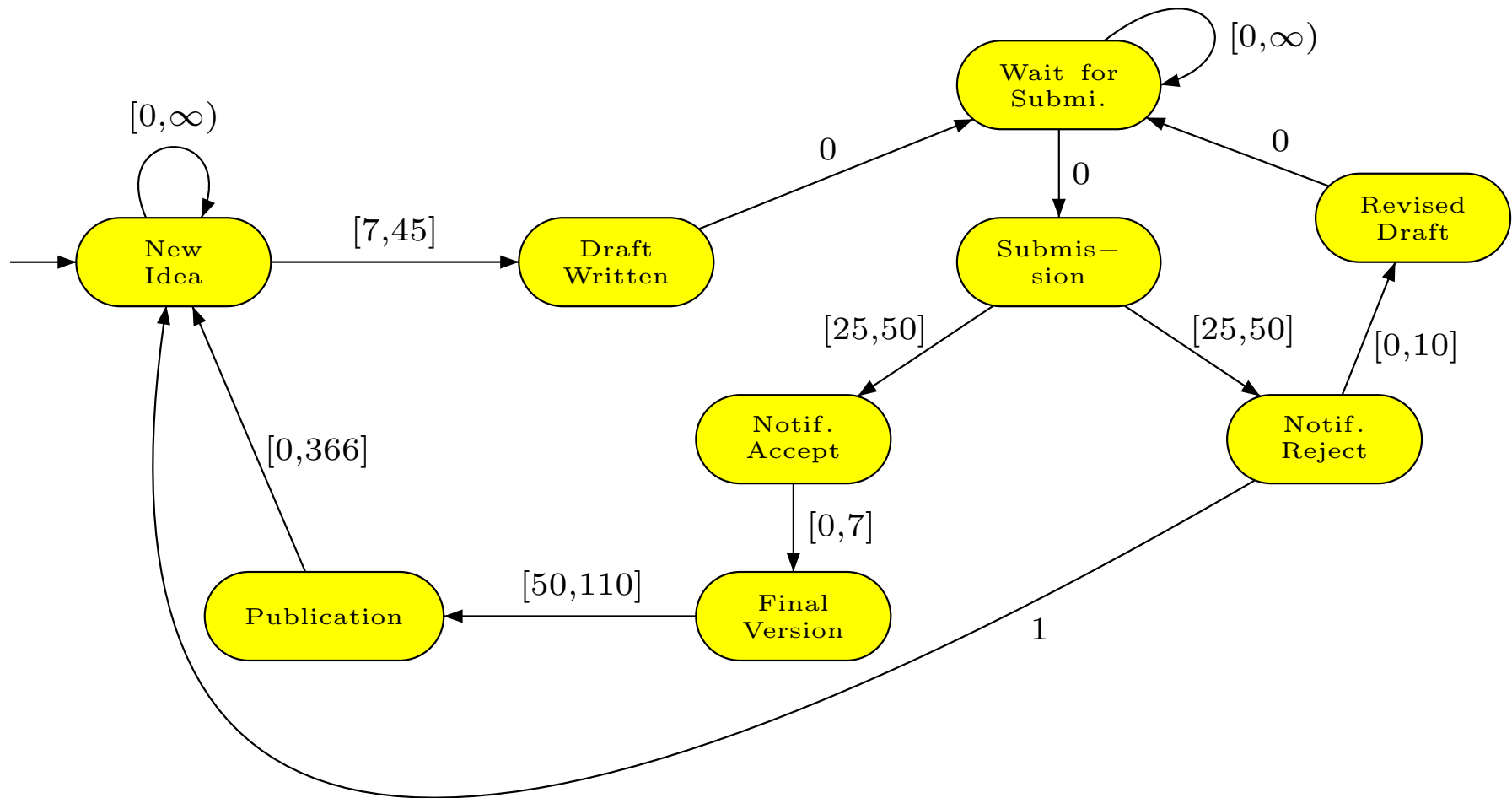
Exercise - 1



$AG(\text{New_Idea} \Rightarrow \neg EF_{<100} \text{Publication})$

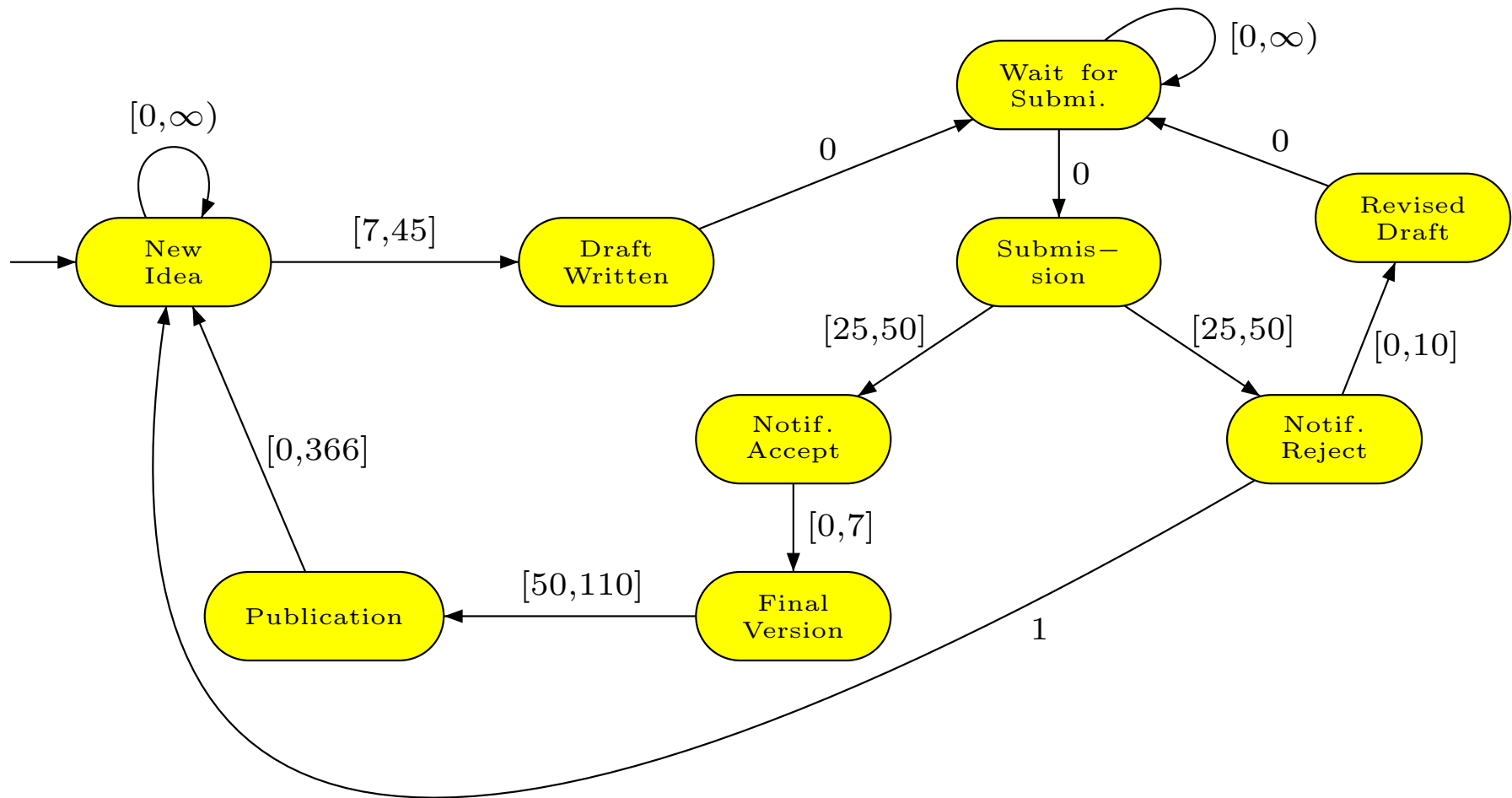
(no !)

Exercise - 2



$AG(\text{New_Idea} \Rightarrow \neg EF_{<60} \text{Publication})$

Exercise - 2



$AG(\text{New_Idea} \Rightarrow \neg EF_{<60} \text{Publication})$

(yes !)

Model-checking $TCTL$

Theorem [EMSS92, LST00]:

Model-checking $TCTL$ over $ssDKS$ can be done in $O(|S|^3 \cdot |\varphi|)$

And over DKS ?

Model-checking $TCTL$

Proposition:

Model-checking formulae of the form $EF_{=c} P$ over DKS is NP-hard.

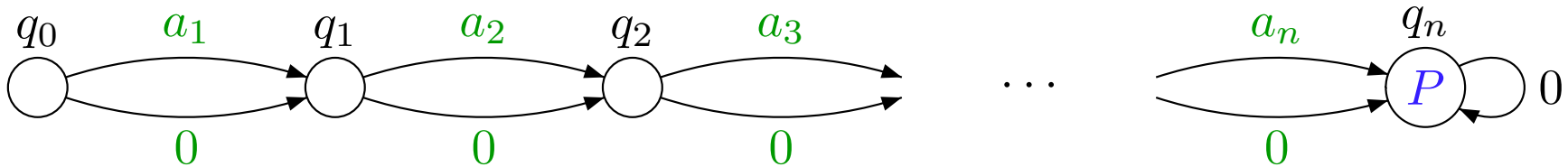
Model-checking $TCTL$

Proposition:

Model-checking formulae of the form $EF_{=c} P$ over DKS is NP-hard.

Reduction from KNAPSACK [GJ79]: given a finite set $A = \{a_1, \dots, a_n\}$ of natural integers, and some target D , is there a subset A' of A s.t.
 $D = \sum_{a \in A'} a$.

This is the case iff $q_0 \models EF_{=D} P$ in the following DKS:



Model-checking $TCTL_{\leq, \geq}$

Let $TCTL_{\leq, \geq}$ denote the fragment of $TCTL$ where equality constraints on modalities are not allowed:

Theorem: Model-checking $TCTL_{\leq, \geq}$ over **DKSs** can be done in time $O(|S|^2 \cdot |\varphi|)$.

Idea of the proof: It is enough to extend the classical CTL algorithm with decision procedures running in time $|S|^2 \cdot \lceil \log c \rceil$ for each modality $E P_1 U_{\sim c} P_2$ and $A P_1 U_{\sim c} P_2$.

Model-checking $TCTL_{\leq, \geq}$

Decision procedure for $\varphi = E P_1 U_{\leq c} P_2$

- We restrict to the **subgraph** where only states satisfying $E P_1 U P_2$ have been kept, and where we only consider the **left extremity** of intervals on edges.
- Then for every state q we compute the **smallest duration** (call it c_q) of a path from q to some state satisfying P_2
This can be done in time $O(|S^2|)$ using a classical **single-source shortest paths** algorithm.

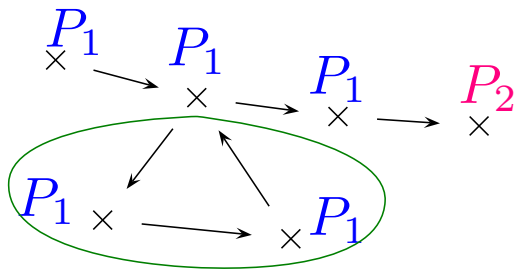
Then $q \models \varphi$ iff $c_q \leq c$.

Model-checking $TCTL_{\leq, \geq}$

Decision procedure for $\varphi = E P_1 U_{\geq c} P_2$

There are 2 ways a state can verify φ :

- a **simple path** (i.e. with **no loop**). Similar to the previous case.
- with loops:



We add a new proposition $P_{SCC+(P_1)}$ labeling every state that belongs to a strongly connected component with duration > 0 , satisfying P_1 .

Then: $q \models E P_1 U_{\geq c} P_2$ if $q \models E P_1 U (P_1 \wedge P_{SCC+(P_1)} \wedge E P_1 U P_2)$

Other operators can be handled in a similar way.

Model-checking $TCTL$

Proposition: Model-checking $TCTL$ over $DKSs$ is in Δ_2^p .

Δ_2^p is the class P^{NP} of problems that can be solved by a deterministic polynomial-time Turing machine that has access to an **NP oracle**.

Model-checking $TCTL$

Proposition: Model-checking $TCTL$ over DKSs is in Δ_2^p .

Δ_2^p is the class P^{NP} of problems that can be solved by a deterministic polynomial-time Turing machine that has access to an NP oracle.

This proposition is based on the following lemma:

Lemma:

Model-checking formulae of the form $E P_1 U_{=c} P_2$ or $A P_1 U_{=c} P_2$ over DKS is in NP.

Model-checking $E P_1 U_{=c} P_2$

$$q_0 \models E P_1 U_{=c} P_2 \Leftrightarrow \exists \pi = q_0 \xrightarrow{d_1} q_1 \xrightarrow{d_2} \dots \xrightarrow{d_n} q_n \text{ such that } \begin{cases} q_i \models P_1 \\ q_n \models P_2 \\ \sum d_i = c \\ n < c \cdot |Q| \end{cases}$$

Model-checking $\mathbf{E} P_1 \mathbf{U}_{=c} P_2$

$$q_0 \models \mathbf{E} P_1 \mathbf{U}_{=c} P_2 \Leftrightarrow \exists \pi = q_0 \xrightarrow{d_1} q_1 \xrightarrow{d_2} \dots \xrightarrow{d_n} q_n \text{ such that } \begin{cases} q_i \models P_1 \\ q_n \models P_2 \\ \sum d_i = c \\ n < c \cdot |Q| \end{cases}$$

Let $\Phi_\pi: R \rightarrow \mathbb{N}$ be the Parikh image of π , i.e. the number of times each transition is used in π .

- Φ_π can be encoded in polynomial size
- Given $\Phi: R \rightarrow \mathbb{N}$, we can decide in polynomial time whether Φ corresponds to a witness of $q_0 \models \mathbf{E} P_1 \mathbf{U}_{=c} P_2$.

(Euler circuit Theorem + verification of propositions + verification of the length)

Verifying $\mathbf{E} P_1 \mathbf{U}_{=c} P_2$ can be done in NP.

Complexity of *TCTL* model-checking

Theorem:

Model-checking *TCTL* over *DKS* is Δ_2^p -complete.

The proof of Δ_2^p -hardness is done by reduction of *SNSAT* (“*sequentially nested SAT*”) to *TCTL* model checking.

A Δ_2^p -complete problem

SNSAT (*sequentially nested SAT*) is Δ_2^p -complete ([LMS01]):

$$\mathcal{I} = \left[\begin{array}{l} x_1 := \exists Z_1 F_1(Z_1), \\ x_2 := \exists Z_2 F_2(x_1, Z_2), \\ \vdots \\ x_n := \exists Z_n F_n(x_1, \dots, x_{n-1}, Z_n) \end{array} \right]$$

where each F_i is a 3-CNF.

\mathcal{I} defines a unique valuation $v_{\mathcal{I}}$ of the variables in X where:

$v_{\mathcal{I}}(x_i) = \top$ iff $F_i(v_{\mathcal{I}}(x_1), \dots, v_{\mathcal{I}}(x_{i-1}), Z_i)$ is satisfiable.

\rightsquigarrow Deciding whether $v_{\mathcal{I}}(x_n) = \top$.

SNSAT can be reduced to *TCTL* model-checking.

Δ_2^p -hardness of *TCTL* model-checking

Assume $F_i = \bigwedge_l \bigvee_{m=1}^3 \alpha_{i,l,m}$

With every disjunct $\bigvee_m \alpha_{i,l,m}$ we associate a clause $C_{i,l}$ of the form $\overline{x_i} \vee \bigvee_m \alpha_{i,l,m}$.
Let $\mathcal{C}l$ be $\{C_1, \dots, C_r\}$ the the resulting set of clauses.

Fix some $K > 11$. To variables $u \in X \cup Z$ and clauses $C \in \mathcal{C}l$ we assign weights $s(u)$ and $s(C)$ given by:

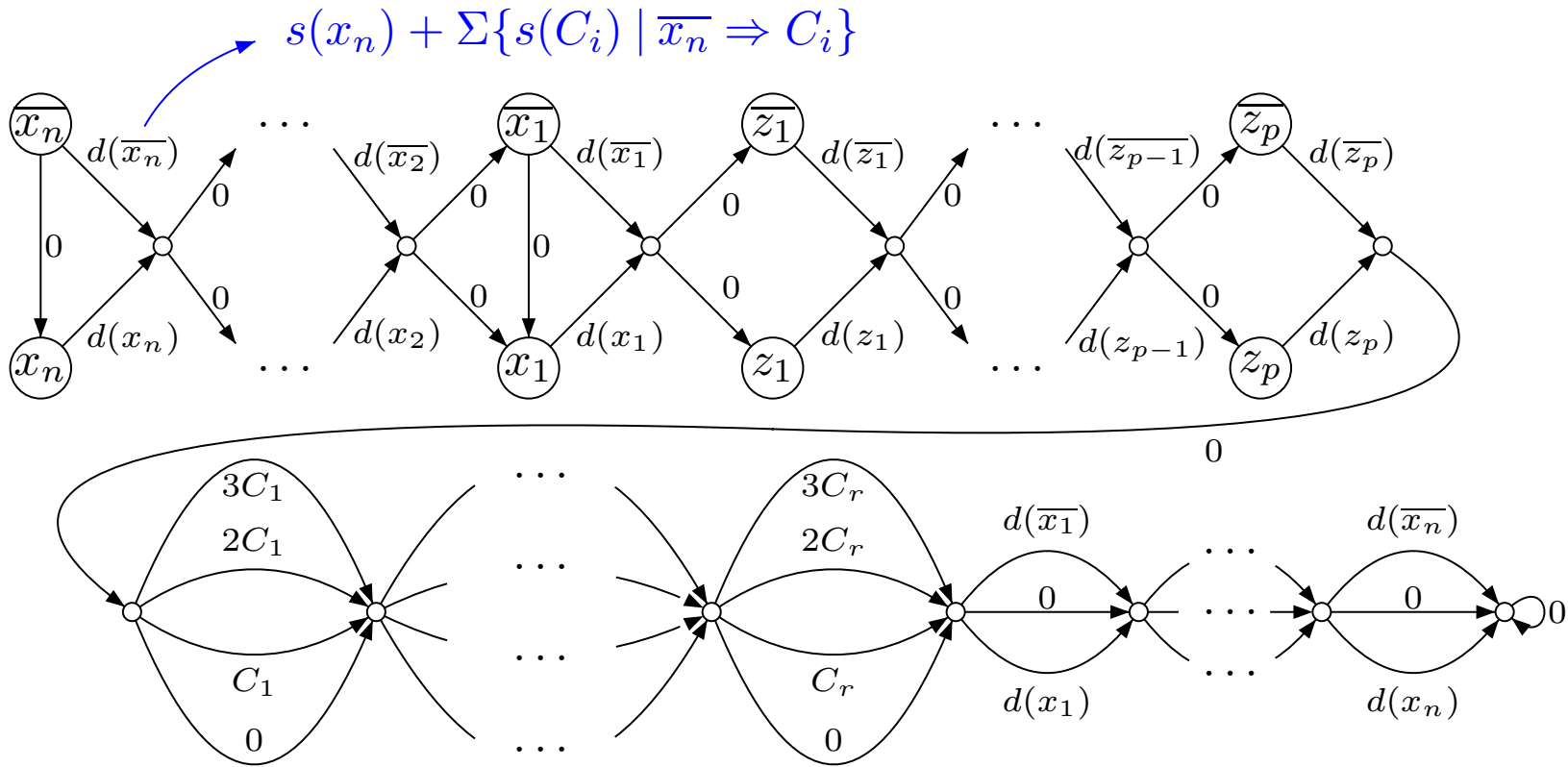
$$s(x_i) = K^i \quad s(z_i) = K^{n+i} \quad s(C_i) = K^{n+p+i} \quad (p = |Z|)$$

A multiset \mathcal{M} of variables and clauses has weight $s(\mathcal{M}) = \sum_x s(x) \times \mathcal{M}(x)$.

Now if $\mathcal{M}(x) < K$ and $\mathcal{M}'(x) < K$ for all $x \in X \cup Z \cup \mathcal{C}l$, then:

$$s(\mathcal{M}) = s(\mathcal{M}') \quad \text{iff} \quad \mathcal{M} = \mathcal{M}'.$$

Δ_2^p -hardness of *TCTL* model-checking



$$v_{\mathcal{I}}(x_n) = \top \text{ iff } x_n \models \varphi_{2n-1} \text{ with: } \varphi_k = \mathbf{E} \left[P_{\overline{x}} \Rightarrow \mathbf{EX} (P_x \wedge \neg \varphi_{k-1}) \right] \mathbf{U}_{=D} \top,$$

$$\varphi_0 = \top, \text{ and } D = \sum_{u \in \text{Var}} s(u) + 4 \times \sum_{C \in \text{Cl}} s(C)$$

Conclusion

	ssDKSs ($\xrightarrow{0/1}$)	tight DKSs (\xrightarrow{n}), DKSs ($\xrightarrow{\rho}$)
$TCTL$ \leq, \geq $\leq, \geq, =$	PTIME-complete	PTIME-complete Δ_2^p -complete
$TLTL$ \leq, \geq $\leq, \geq, =$	PSPACE-complete EXPSPACE-complete	
$TCTL^*$ \leq, \geq $\leq, \geq, =$	PSPACE-complete EXPSPACE-complete	
$TCTL^+$ \leq, \geq $\leq, \geq, =$	Δ_2^p -complete	
$TCTL_c$	PSPACE-complete	

1. Exact durations make model-checking harder.
2. Polynomial time model-checking is possible if one considers $TCTL_{\leq, \geq}$ **or** $TCTL$ and ssDKSs
3. A new Δ_2^p -complete model-checking problem.
4. We are considering other semantics.

References - 1

- [ACD93] R. Alur, C. Courcoubetis, and D. Dill. Model-checking in dense real-time. *Information and Computation*, 104(1):2–34, 1993.
- [AH92] R. Alur and T. A. Henzinger. Logics and models of real time: A survey. In *Real-Time: Theory in Practice, Proc. REX Workshop, Mook, NL, June 1991*, volume 600 of *Lecture Notes in Computer Science*, pages 74–106. Springer, 1992.
- [AH94] R. Alur and T. A. Henzinger. A really temporal logic. *Journal of the ACM*, 41(1):181–203, 1994.
- [AL99] L. Aceto and F. Laroussinie. Is your model checker on time? In *Proc. 24th Int. Symp. Math. Found. Comp. Sci. (MFCS'99), Szklarska Poreba, Poland, Sep. 1999*, volume 1672 of *Lecture Notes in Computer Science*, pages 125–136. Springer, 1999.
- [EMSS92] E. A. Emerson, A. K. Mok, A. P. Sistla, and J. Srinivasan. Quantitative temporal reasoning. *Real-Time Systems*, 4(4):331–352, 1992.
- [ET99] E. A. Emerson and R. J. Trefler. Parametric quantitative temporal reasoning. In *Proc. 14th IEEE Symp. Logic in Computer Science (LICS'99), Trento, Italy, July 1999*, pages 336–343. IEEE Comp. Soc. Press, 1999.

References - 2

- [GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability. A Guide to the Theory of NP-Completeness*. Freeman, 1979.
- [LMS01] F. Laroussinie, N. Markey, and Ph. Schnoebelen. Model checking CTL^+ and $FCTL$ is hard. In *Proc. 4th Int. Conf. Foundations of Software Science and Computation Structures (FOSSACS'2001), Genova, Italy, Apr. 2001*, volume 2030 of *Lecture Notes in Computer Science*, pages 318–331. Springer, 2001.
- [LST00] F. Laroussinie, Ph. Schnoebelen, and M. Turuani. On the expressivity and complexity of quantitative branching-time temporal logics. In *Proc. 4th Latin American Symposium on Theoretical Informatics (LATIN'2000), Punta del Este, Uruguay, Apr. 2000*, volume 1776 of *Lecture Notes in Computer Science*, pages 437–446. Springer, 2000. Journal version as [\[LST02\]](#).
- [LST02] F. Laroussinie, Ph. Schnoebelen, and M. Turuani. On the expressivity and complexity of quantitative branching-time temporal logics. *Theoretical Computer Science*, 2002. To appear.