A Parametric Model Checking Approach for Real-Time Systems Design

Chaiwat Sathawornwichit

Katayama Lab.
School of Information Science

JAIST
JAPAN ADVANCED INSTITUTE OF SCIENCE AND TECHNOLOGY
1990

Outline

- Background
- Parametric Model Checking
- Problems and our Approach
- Parametric Timed System Model
  - Parametric Timed Structure (PTS)
- Parametric Timed Temporal Logic
  - Parametric CTL (PARCTL)
- Deriving Parametric Condition
- Determining Optimal Factor by Constraint Solving
- Discussion & Conclusion

Background

- The majority of computer system today are real-time systems
  - Embedded in devices.
  - Running infrastructure control applications.
  - Our society relies so much on them
- Timing characteristic is a crucial aspect of safety
  - *Correct* **action** must be taken at the *right* **time**.
- Formal verification techniques have been developed for assuring the correctness of real-time systems
  - Model checking for real-time systems.

Background (2)

- Aspects of time make model checking approach for real-time systems seriously complicated.
  - *Time* is introduced to the **model**, and the **temporal logic**.
  - Correct action sequences **+** Correct timing.
- Timed model
  - Timed transition graph (a.k.a. timed Kripke structure)
    - Time duration in the transitions
    - Simple (can be model checked in linear to model size)
  - Timed automata
    - Automata + clocks
    - Transition conditions on clock values
    - Clocks can be set / reset
    - Very complicated (complexity depends on #clocks)

Parametric Model Checking and Problems

- Parametric Model Checking
  - Abstraction of time values by variables.
  - The use of variables in
    - Temporal logic formulas, and
    - Timed models
- The Problems
  - Determine whether *there exists a valuations* of parameters under which the model *M* satisfies the property *p*.
  - Compute the *solution set of parameters* under which the model *M* satisfies the property *p*.

The Difficulties

- For time automata
  - Very high *computation complexity*, inapplicable to large problem.
  - *Undecidable* when #clocks ≧ 3.
- For timed transition graph
  - Only the use of parameters in temporal logic has been introduced so far.
  - Parametric model for timed transition graph has not been studied.

Our Approaches

- Instead of computing the solution set of parameters, we derive the parametric conditions over parameters (as a system of linear inequalities).
- We develop this approach for timed transition graph.
- We further propose the application of mathematical tools with this approach for determining the design for solution for an optimal criteria.

Contributions

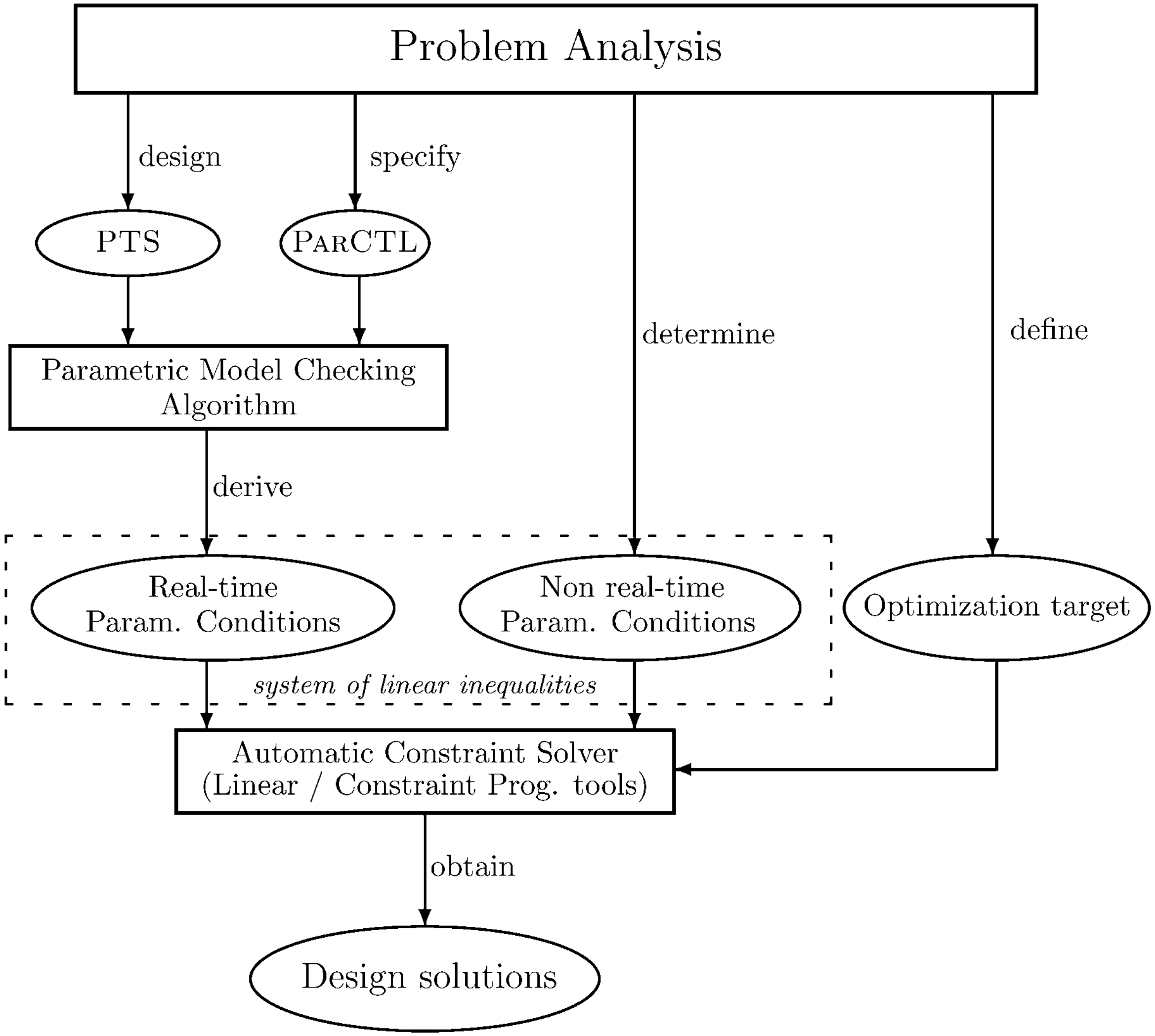
(1) Introduce parameters to timed transition graph model.

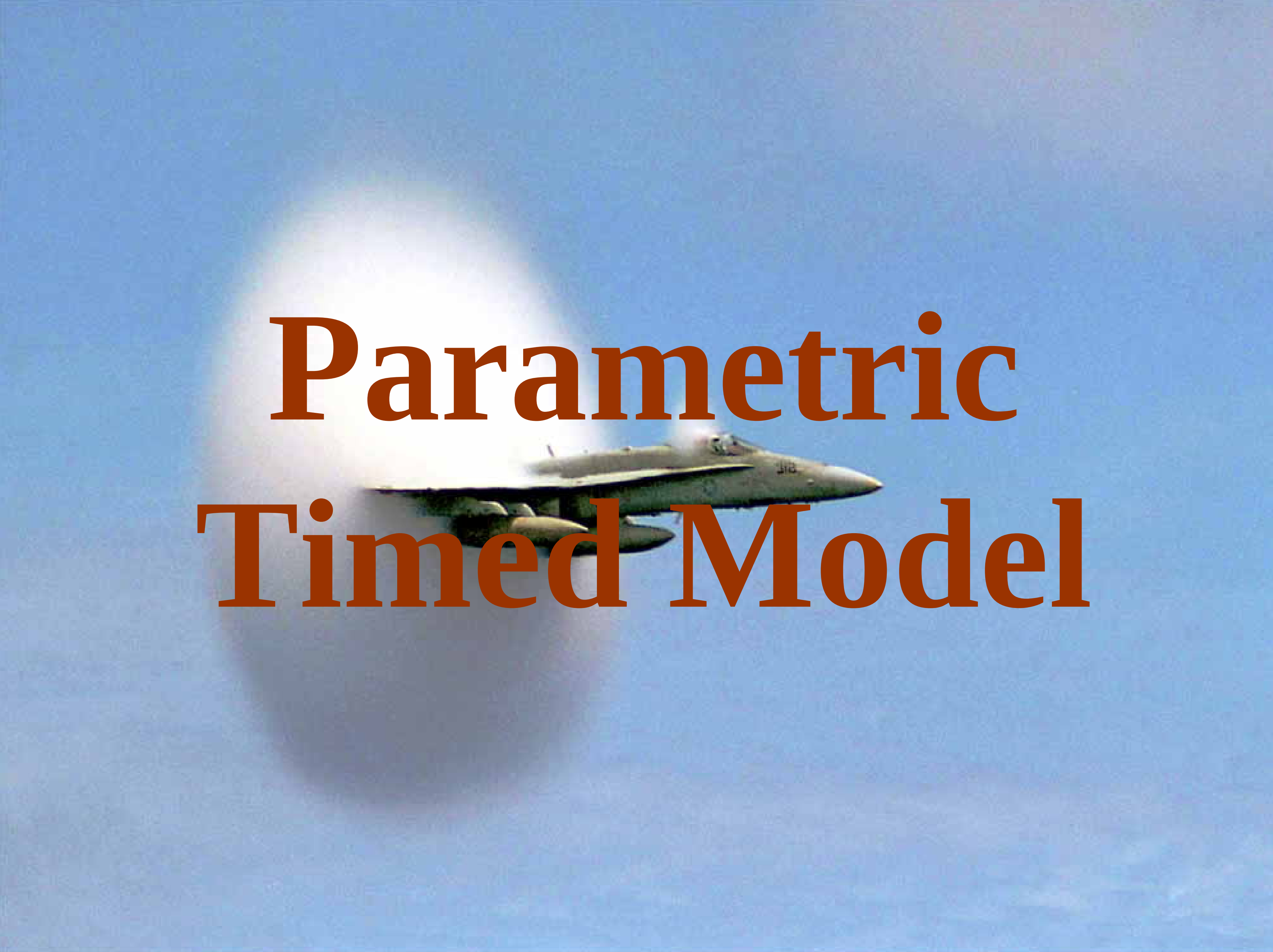
(2) Define a parametric timed temporal logic for reasoning real-time properties over (1).

(3) Provide algorithms for deriving parameter conditions satisfying real-time property and non real-time restriction, e.g. cost, development time.

(4) Demonstrate the application of mathematical programming methods to determine the parameter values which optimize a particular objective.

Parametric Approach Framework





Parametric Timed Model

Parametric Timed Structure (PTS)

- Non-deterministic finite state machine
  - With time durations labelled on transitions.
  - The durations can be linear combinations of parameters
- Extension of
  - Simply-timed model [Markey et al. 2004],
  - Timed Kripke structure [Emerson & Trefler 1999],
  - Timed transition graph [Campos & Clarke 1994].

Syntax of PTS

A *parametric timed structure* $\mathcal{M} = (S, S_0, \vec{x}, T, L)$ consists of

S	A finite set of states.
$S_0 \subseteq S$	A set of initial states.
$\vec{x}$	A finite vector of real-valued time variables.
$\overline{X}$	The set of linear expressions over $\vec{x}$.
$T \subseteq S \times \overline{X} \times S$	A finite set of parametric transition relation.
$L : S \rightarrow 2^{AP}$	A labeling function which assigns to each state the set of atomic propositions hold in the state.

$$L(s) = \{f \mid f \in AP \wedge s \models f\}$$

Syntax of PTS

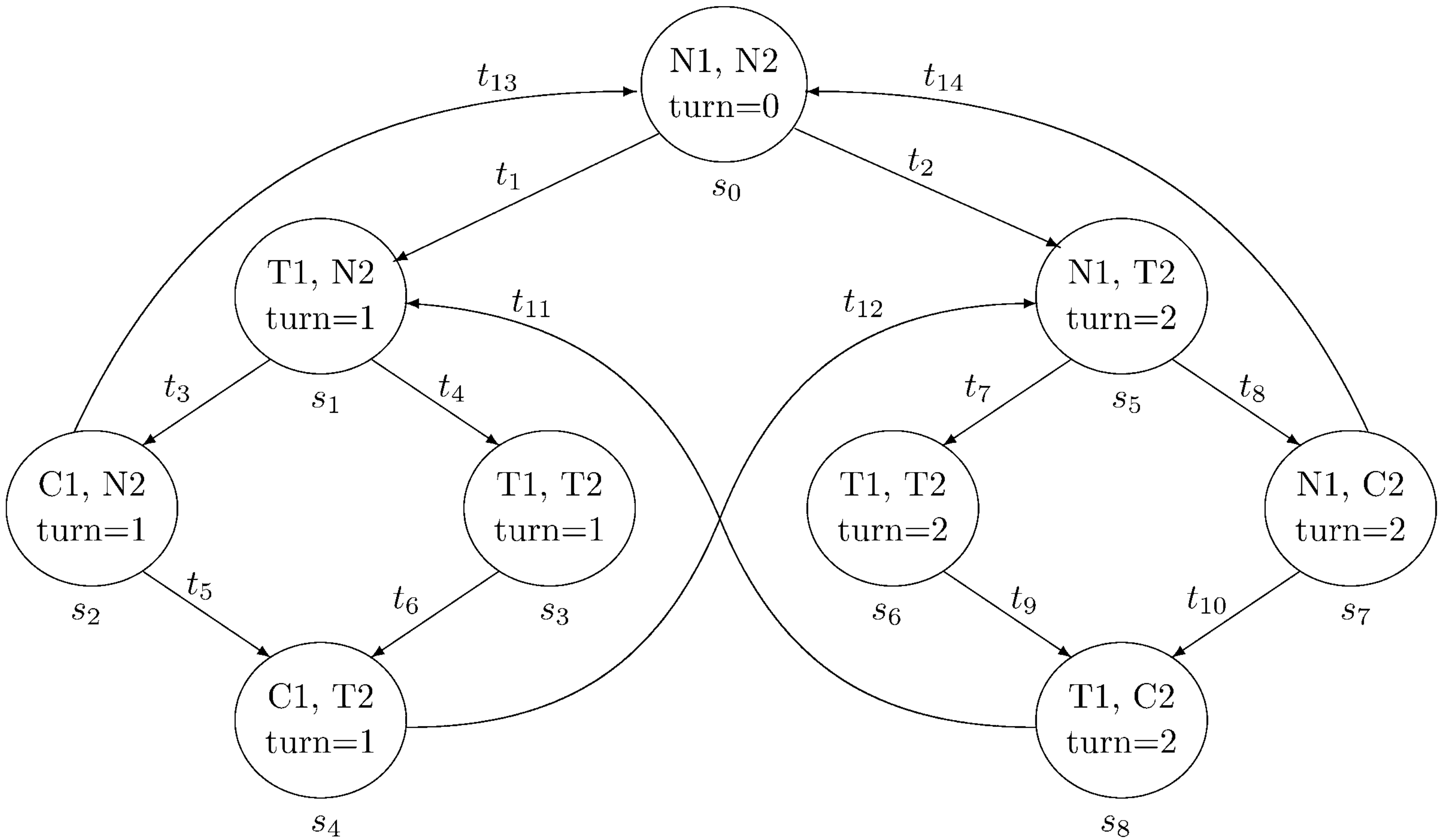
A PTS $\mathcal{M}$ with time variable vector $\vec{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n_{0+}$

- The size n of $\vec{x}$ is the degree of $\mathcal{M}$
- A *linear expression* over $\vec{x}$ is of the form,

$$\sum_{i=1}^{n} c_i x_i + c$$

- When all $c_i = 0$, the expression become a constant c.

PTS Examples



A mutual exclusion protocol example

Basic Constructs

Constraint, Condition, and Predicate

- A *linear constraint* over $\vec{x}$ is a combination of the form $(\alpha \sim \beta)$
  - α, β are linear expression
  - $\sim \in \{<, \leq, =, \geq, >\}$.
  - Ex., $x_1 + x_2 \leq 5$ is a constraint over (x_1, x_2)
- A *linear condition* over $\vec{x}$ is a finite *conjunction* of linear constraints over $\vec{x}$.
  - ex. $(x_1 + x_2 \leq 5 \ \wedge \ x_1 \leq 5 \ \wedge \ x_3 \geq 2x_1)$
- A *linear predicate* over $\vec{x}$ is a finite *disjunction* of linear conditions over $\vec{x}$.
  - ex. $(x_1 + x_2 \leq 5) \vee (x_1 \leq 5 \ \wedge \ x_3 \geq 2x_1)$

Assignment and Evaluation

For a *linear expression* e with parameters $\vec{x} = (x_1, x_2)$

- $e_{[\vec{x} \leftarrow \vec{v}]}$ is an evaluation of e by assignment $\vec{v}$
  - ex 1. $e = 2x_1 + x_2$; $e_{[\vec{x} \leftarrow (1,2)]} = 2 \cdot 1 + 2 = 4$
  - ex 2. $e = 2x_1 + x_2$; $e_{[\vec{x} \leftarrow (1,x_2)]} = 2 + y$
- $e_{[\vec{v}]}$ abbreviates $e_{[\vec{x} \leftarrow \vec{v}]}$
- For a linear expression, evaluation results in another linear expression.

Assignment and Evaluation

For a *linear condition* or *linear predicate* q with $\vec{x} = (x_1, x_2)$

- $q_{[\vec{x} \leftarrow \vec{v}]}$ is an evaluation of q by assignment $\vec{v}$
  - ex 1.
    $$q = (x_1 + x_2 \ \geq \ 2x_2 + 1);$$
    $$q_{[(x_1,2)]} = (x_1 + 2 \geq 2 \cdot 2 + 1)$$
    $$x_1 \geq 3$$
  - ex 2.
    $$q_{[(2,1)]} = (2 + 1 \geq 2 \cdot 1 + 1)$$
    $$3 \geq 3$$
  - ex 3.
    $$q' = (x_2 \ \geq \ x_1 + 1);$$
    $$q_{[(x_1,2)]} = (2 \geq x_1 + 1)$$
    $$1 \geq x$$
  - ex 4. $q'' = (q \wedge q');\ q''_{[(x_1,2)]} = (x_1 \geq 3 \ \wedge \ 1 \geq x) = \text{False}$

Assignment and Evaluation

- For linear predicate, evaluations results in another predicate which equivalent to a region in $\mathbb{R}^n_{0+}$)

- Evaluations in the previous examples result in:

  1. region $(x \geq 3 \ \wedge \ y = 2)$
  2. region (a point at $(x = 2 \ \wedge \ y = 1)$)
  3. region $(x \leq 1 \ \wedge \ y = 2)$
  4. empty region (intersection of region in 1. and 3. $= \emptyset$)

Linear Predicate and Assignment

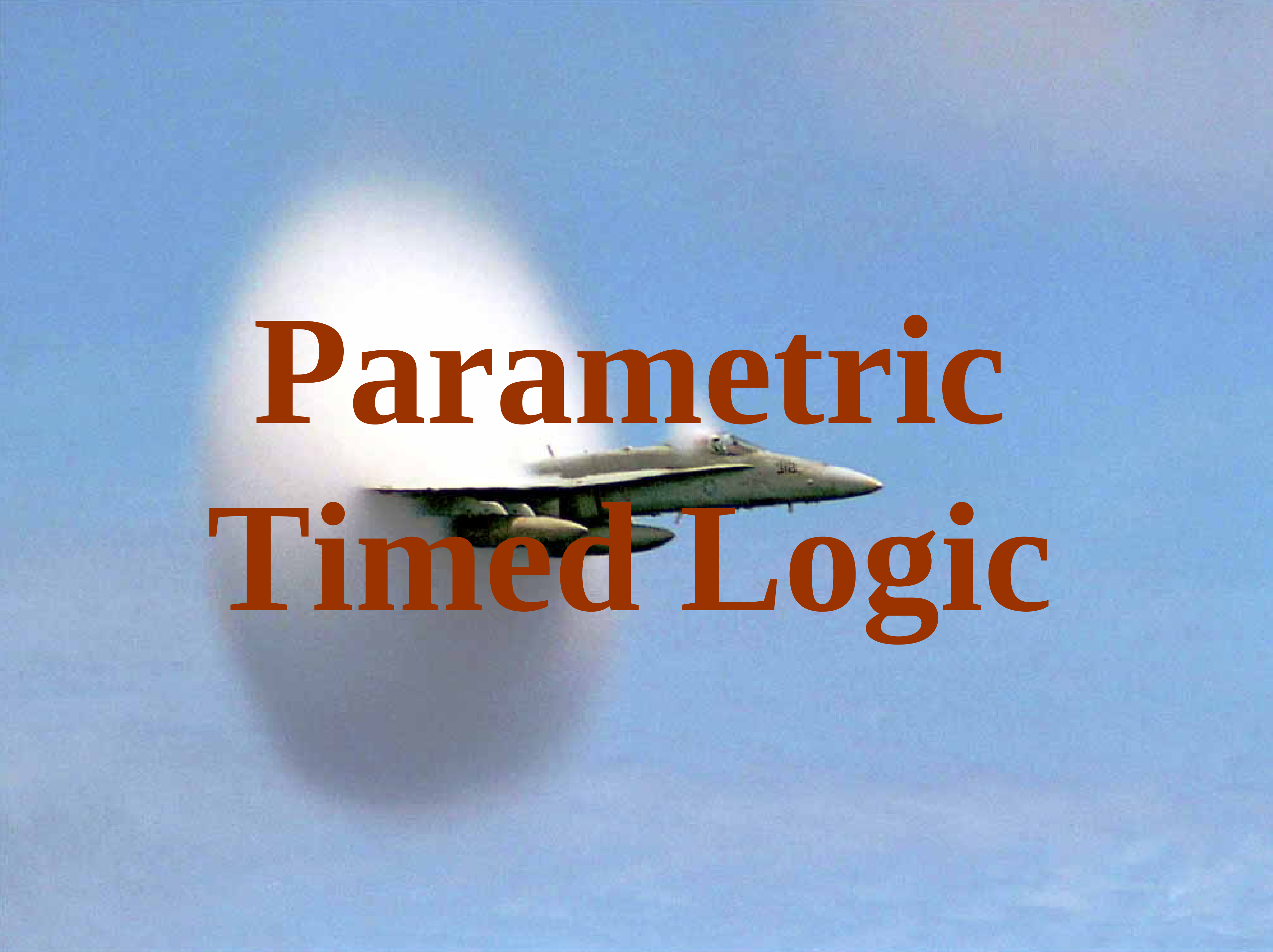
Let $\mathtt{C}$ be a linear condition which is a conjunction of linear constraints $\mathtt{c} \in Q$

- $[\![\mathtt{C}]\!]$ denotes a set of assignment (which is region in $\mathbb{R}^n_{0+}$)
- Such that, any assignment v in $[\![\mathtt{C}]\!]$ satisfies the predicate $\mathtt{C}$ ($v \models \mathtt{C}$).
- That is for an assignment $\vec{v}$:

$$\vec{v} \in [\![\mathtt{C}]\!] \quad \text{iff} \quad q_{[\vec{x} \leftarrow \vec{v}]} \neq \emptyset$$

- The assignment set $[\![\mathtt{C}]\!]$ is determined by $[\![\mathtt{C}]\!] = \bigcap_{\mathtt{c} \in Q} [\![\mathtt{c}]\!]$

Parametric Timed Logic



Syntax of Parametric CTL(ParCTL)

ParCTL formulas inductively defined by the grammar

$$
\begin{aligned}
f \quad ::= \quad & p \mid \neg f \mid f \wedge f \mid f \vee f \\
& \mid f \; \mathsf{EU}^{\sim \alpha} f \mid f \; \mathsf{AU}^{\sim \alpha} f
\end{aligned}
$$

- $\alpha \in \overline{X}$: a linear expression $(\sum_i c_i x_i + c)$
- $\sim \in \{<, \leq, =, \geq, >\}$
- Now, we consider only $<$ and $\leq$ cases.
- $p \in AP$: an atomic proposition

Semantics of ParCTL

- $s \models_{\vec{v}} p$ iff $p \in L(s)$
- $s \models_{\vec{v}} \neg f$ iff $s \not\models_{\vec{v}} f$
- $s \models_{\vec{v}} f_1 \wedge f_2$ iff $s \models_{\vec{v}} f_1$ and $s \models_{\vec{v}} f_2$
- $s \models_{\vec{v}} f_1 \vee f_2$ iff $s \models_{\vec{v}} f_1$ or $s \models_{\vec{v}} f_2$
- $s \models_{\vec{v}} f_1 \ \mathsf{EU}^{\sim\alpha} f_2$ iff there exists a path $\pi \in \Pi(s)$, $i, j \in \mathbb{N}$ such that

$$\exists i. \Big[\big(s_{(i)} \models_{\vec{v}} f_2\big) \wedge \big(\lambda(\pi, i)[\vec{v}] \sim \alpha[\vec{v}]\big) \wedge \forall j < i. [(s_{(j)} \models_{\vec{v}} f_1) \wedge (s_{(j)} \not\models_{\vec{v}} f_2)] \Big]$$

- $s \models_{\vec{v}} f_1 \ \mathsf{AU}^{\sim\alpha} f_2$ iff for all paths $\pi \in \Pi(s)$, $i, j \in \mathbb{N}$, such that

$$\exists i. \Big[\big(s_{(i)} \models_{\vec{v}} f_2\big) \wedge \big(\lambda(\pi, i)[\vec{v}] \sim \alpha_{[\vec{v}]}\big) \wedge \forall j < i. [(s(j) \models_{\vec{v}} f_1) \wedge (s_{(j)} \not\models_{\vec{v}} f_2)] \Big]$$

Semantics of ParCTL (2)

- For a path $\pi = s_0 \xrightarrow{x_0} s_1 \xrightarrow{x_1} s_2 \cdots \xrightarrow{x_{i-1}} s_i \cdots$ in $\mathcal{M}$
- $\lambda(\pi, i)$ denotes *duration function*:

$$\lambda(\pi, i) \stackrel{\text{def}}{=} \sum_{j=0}^{i-1} e_j$$

Derivation of Parametric Condition

Parametric Condition Derivation

- *Parametric condition* $\mathcal{P}$ is a linear condition over parameters $\vec{x}$ of a PTS $\mathcal{M}$ to satisfy a PARCTL property.

- $\mathcal{P}$ defines a set of assignments

$$[\![\mathcal{P}(s,f)]\!] \stackrel{\text{def}}{=} \{\vec{v} \mid s \models_{\vec{v}} f\}$$

such that, any assignment $\vec{v}$ in $[\![\mathcal{P}(s,f)]\!]$

$$\vec{v} \models \mathcal{P}(s,f) \text{ iff } \mathcal{M}, s \models_{\vec{v}} f$$

Parametric Predicate

Parametric predicate $\mathcal{P}(s, f)$ is compute inductive on the subformula of f.

$$\mathcal{P}(s,p) \quad := \quad \mathcal{M}, s \models p$$

$$\mathcal{P}(s,\neg f) \quad := \quad \neg\mathcal{P}(s,f)$$

$$\mathcal{P}(s,f_1 \wedge f_2) \quad := \quad \mathcal{P}(s,f_1) \wedge \mathcal{P}(s,f_2)$$

$$\mathcal{P}(s,f_1 \vee f_2) \quad := \quad \mathcal{P}(s,f_1) \vee \mathcal{P}(s,f_2)$$

$$\mathcal{P}(s,f_1 \ \mathsf{AU}^{\sim\alpha} f_2) \quad := \quad \{\mathcal{P}(s,f_2) \wedge (0 \sim \alpha)\} \vee \{\mathcal{P}(s,f_1) \wedge \bigwedge_{(t,s')\in AC(s)} \mathcal{P}(s', f_1 \ \mathsf{AU}^{\sim\alpha-t} f_2)\}$$

$$\mathcal{P}(s,f_1 \ \mathsf{EU}^{\sim\alpha} f_2) \quad := \quad \{\mathcal{P}(s,f_2) \wedge (0 \sim \alpha)\} \vee \{\mathcal{P}(s,f_1) \wedge \bigvee_{(t,s')\in AC(s)} \mathcal{P}(s', f_1 \ \mathsf{EU}^{\sim\alpha-t} f_2)\}$$

Example: Railroad Crossing Gate

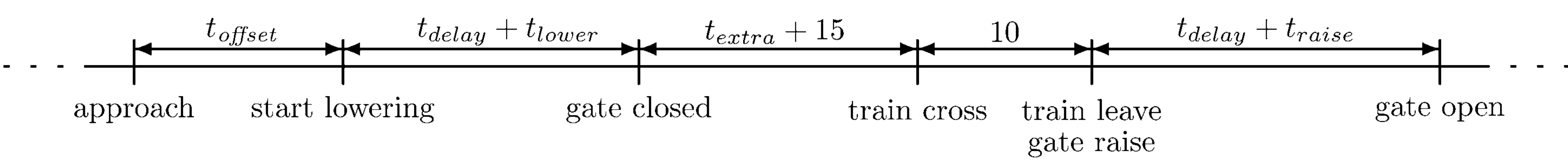
Gate control system:

- 1 controller, and
- 2 gates.
- Minimize the cost,

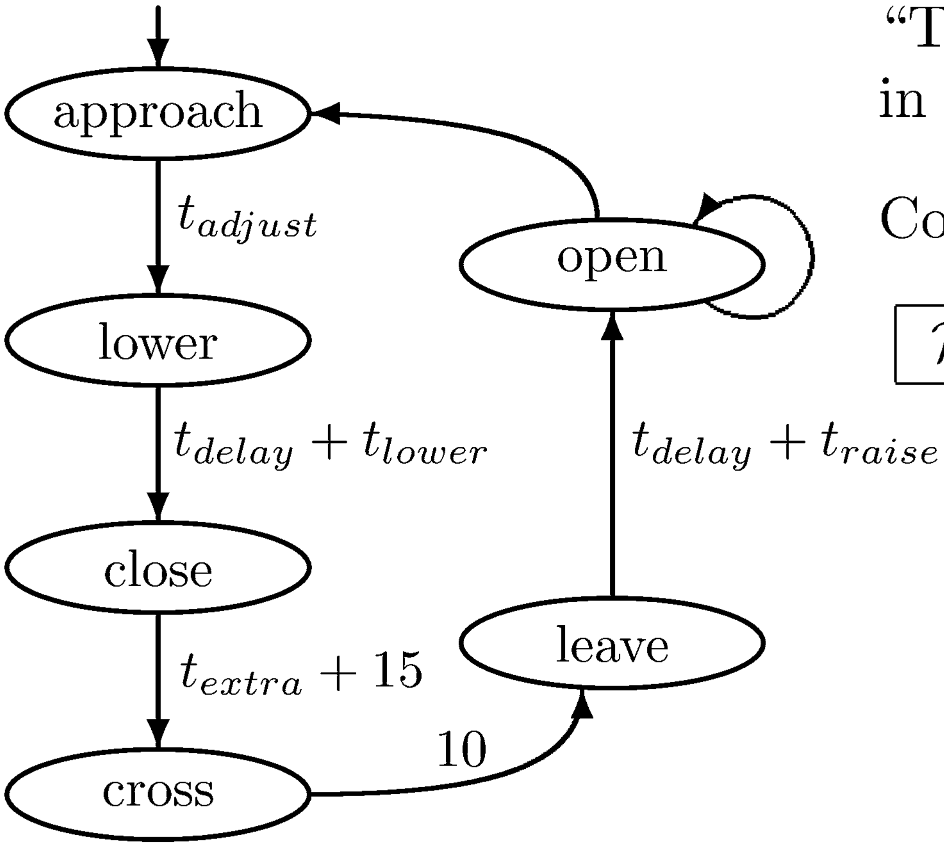
$$\boxed{total\ cost = controller\ cost + 2 \times gate\ cost}$$

controller	t_{offset}	t_{extra}	cost
c_1	4	6	2,000
c_2	8	2	3,000

gate	t_{lower}	t_{delay}	t_{raise}	cost
g_1	12	8	16	400
g_2	6	5	12	600
g_3	4	3	10	800



Example: Railroad Crossing Gate



"The gate must be open again in x seconds after it is lowered."

Compute the parametric condition.

$$\mathcal{P}(s_0, \mathsf{AG}\ (lower \Rightarrow \mathsf{AF}^{\leq x} open))$$

The PTS for railroad crossing gate controller system.

Example: Bridge Crossing Problem

Applying the algorithm, we obtain the parametric condition:

$$t_{lower} + t_{extra} + t_{raise} + 2\, t_{delay} \leq x - 25$$

If the requirements for a crossing are settled that the waiting time x should be less than one minute. The condition becomes:

$$t_{lower} + t_{extra} + t_{raise} + 2\, t_{delay} \leq 35$$

Using the condition with information from the previous tables as input to a linear programming solver (LP_solve).

The minimum cost is choosing c_1 and g_2
$min\ cost = 2000 + (2 \times 600) = 3200$

Disscusion

- The same parametric condition for the gate controller system is applicable to controller systems at different locations by just adapt some parameters or cost factors.
- We implemented the algorithm by graph-based representation in Java.
- The complexity of derivation algorithm is linear to the PTS model size.
- The existing linear programming / integer programming solvers are sophisticated and able to solve large system of inequality as many as hundreads thousands inequalities.

Conclusion

(1) Introduce parameters to timed transition graph ⇒PTS.

(2) Define a parametric timed temporal logic ⇒ ParCTL for reasoning real-time properties over PTS.

(3) Provide algorithms for deriving parametric conditions satisfying real-time property and non real-time constraints.

(4) Demonstrate the application of mathematical programming methods to determine the parameter values which optimize a particular criteria.