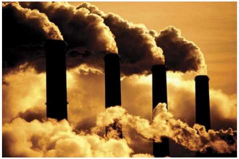
Green Scheduling

Ashutosh Trivedi

Department of Computer Science and Engineering,
IIT Bombay

CS620: New Trends in IT: Modeling and Verification of Cyber-Physical Systems
(26 July 2013)

Peak Demand Reduction in Energy Usage

1. Absence of bulk energy storage technology
2. Base-load vs peaking power plants
3. Energy peaks are expensive:
   - For environment (peaking power plants are typically fossil-fueled)
   - For energy providers
   - For customers (peak power pricing)
4. Energy peaks are often avoidable:
   - Extreme weather and energy peaks
   - Heating, Ventilation, and Air-conditioning (HVAC) Units
5. Load-balancing methods:
   - Load shedding
   - Load shifting
   - Green scheduling [NBPM11]

Green Scheduling

Zones \ HVAC Units Modes	HIGH	LOW	OFF
X (Temp. Change Rate/ Energy Usage)	-2/3	-1/2	2/0.2
Y (Temp. Change Rate/ Energy Usage)	-2/3	-1/2	3/0.2

Green Scheduling

Zones \ HVAC Units Modes	HIGH	LOW	OFF
X (Temp. Change Rate/ Energy Usage)	-2/3	-1/2	2/0.2
Y (Temp. Change Rate/ Energy Usage)	-2/3	-1/2	3/0.2

- Assume that comfortable temperature range is $65^o F$ to $70^o F$.
- Energy is extremely expensive if peak demand dips above 4 units in a billing period

Green Scheduling

Zones \ HVAC Units Modes	HIGH	LOW	OFF
X (Temp. Change Rate/ Energy Usage)	-2/3	-1/2	2/0.2
Y (Temp. Change Rate/ Energy Usage)	-2/3	-1/2	3/0.2

- Assume that comfortable temperature range is $65^o F$ to $70^o F$.
- Energy is extremely expensive if peak demand dips above 4 units in a billing period

Problem

Find an "implementable" switching schedule that keeps the temperatures within comfort zone and peak usage within 4 units?

Green Scheduling: Contd

$\dot{x} = -2$, $\dot{y} = -2$	$\dot{x} = -2$, $\dot{y} = -1$	$\dot{x} = -1$, $\dot{y} = -2$
$M_{H,H}(6)$	$M_{H,L}(5)$	$M_{L,H}(5)$
$\dot{x} = -2$, $\dot{y} = 3$	$\dot{x} = -1$, $\dot{y} = -1$	$\dot{x} = -1$, $\dot{y} = 3$
$M_{H,O}(3.2)$	$M_{L,L}(4)$	$M_{L,O}(2.2)$
$\dot{x} = 2$, $\dot{y} = -2$	$\dot{x} = 2$, $\dot{y} = -1$	$\dot{x} = 2$, $\dot{y} = 3$
$M_{O,H}(3.2)$	$M_{O,L}(2.2)$	$M_{O,O}(0.4)$

Green Scheduling: Contd

$\dot{x} = -2$, $\dot{y} = -2$	$\dot{x} = -2$, $\dot{y} = -1$	$\dot{x} = -1$, $\dot{y} = -2$
$M_{H,H}(6)$	$M_{H,L}(5)$	$M_{L,H}(5)$
$\dot{x} = -2$, $\dot{y} = 3$	$\dot{x} = -1$, $\dot{y} = -1$	$\dot{x} = -1$, $\dot{y} = 3$
$M_{H,O}(3.2)$	$M_{L,L}(4)$	$M_{L,O}(2.2)$
$\dot{x} = 2$, $\dot{y} = -2$	$\dot{x} = 2$, $\dot{y} = -1$	$\dot{x} = 2$, $\dot{y} = 3$
$M_{O,H}(3.2)$	$M_{O,L}(2.2)$	$M_{O,O}(0.4)$

Green Scheduling: Contd

m_1: $\dot{x} = -2$, $\dot{y} = 3$

m_2: $\dot{x} = -1$, $\dot{y} = -1$

m_3: $\dot{x} = -1$, $\dot{y} = 3$

m_4: $\dot{x} = 2$, $\dot{y} = -2$

m_5: $\dot{x} = 2$, $\dot{y} = -1$

m_6: $\dot{x} = 2$, $\dot{y} = 3$

Safe Schedulability Problem

Does there exist a switching schedule using these modes such that the temperatures of all zones stays in comfortable region?

Multi-mode Systems: Safe Schedulability

m_1: $\dot{x} = -2$, $\dot{y} = 3$

m_2: $\dot{x} = -1$, $\dot{y} = -1$

m_3: $\dot{x} = -1$, $\dot{y} = 3$

m_4: $\dot{x} = 2$, $\dot{y} = -2$

m_5: $\dot{x} = 2$, $\dot{y} = -1$

m_6: $\dot{x} = 2$, $\dot{y} = 3$

Safe set: $x \in [65, 70], y \in [65, 70]$

s_0: $x = 68$, $y = 68$

Keywords: State, Schedule, periodic schedule, ultimately periodic schedule, trajectory, and safe schedule

Multi-mode Systems: Safe Schedulability

m_1: $\dot{x} = -2$, $\dot{y} = 3$

m_2: $\dot{x} = -1$, $\dot{y} = -1$

m_3: $\dot{x} = -1$, $\dot{y} = 3$

m_4: $\dot{x} = 2$, $\dot{y} = -2$

m_5: $\dot{x} = 2$, $\dot{y} = -1$

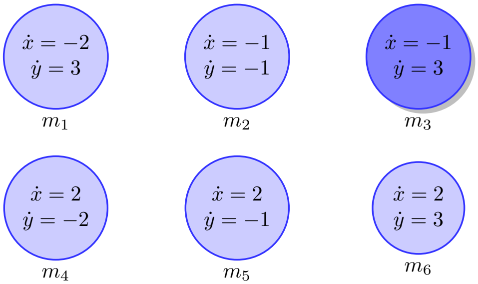
m_6: $\dot{x} = 2$, $\dot{y} = 3$

Safe set: $x \in [65, 70], y \in [65, 70]$

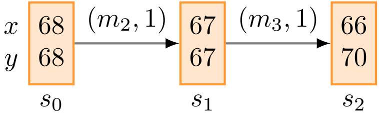
$$\begin{array}{c} s_0 \\ x: 68 \\ y: 68 \end{array} \xrightarrow{(m_2, 1)} \begin{array}{c} s_1 \\ x: 67 \\ y: 67 \end{array}$$

Keywords: State, Schedule, periodic schedule, ultimately periodic schedule, trajectory, and safe schedule

Multi-mode Systems: Safe Schedulability

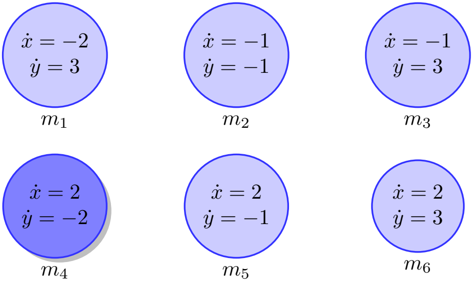


Safe set: $x \in [65, 70], y \in [65, 70]$

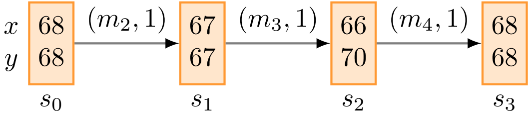


Keywords: State, Schedule, periodic schedule, ultimately periodic schedule, trajectory, and safe schedule

Multi-mode Systems: Safe Schedulability

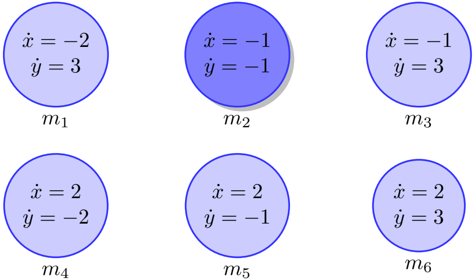


Safe set: $x \in [65, 70], y \in [65, 70]$

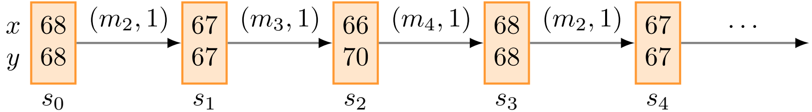


Keywords: State, Schedule, periodic schedule, ultimately periodic schedule, trajectory, and safe schedule

Multi-mode Systems: Safe Schedulability

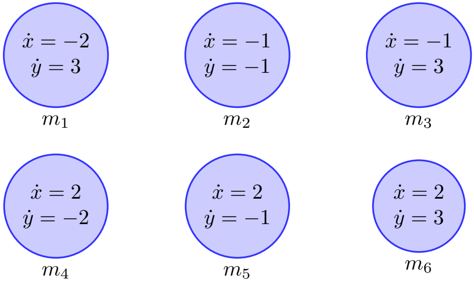


Safe set: $x \in [65, 70], y \in [65, 70]$

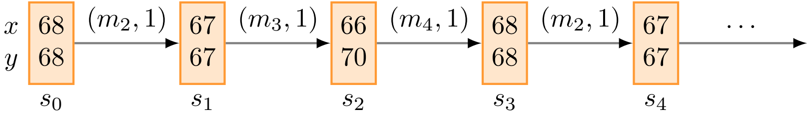


Keywords: State, Schedule, periodic schedule, ultimately periodic schedule, trajectory, and safe schedule

Multi-mode Systems: Safe Schedulability

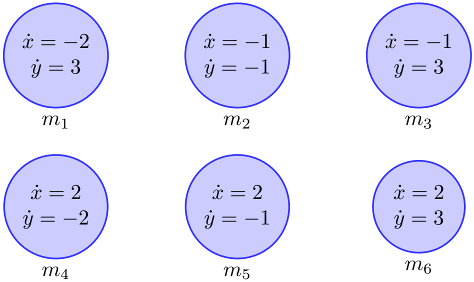


Safe set: $x \in [65, 70], y \in [65, 70]$

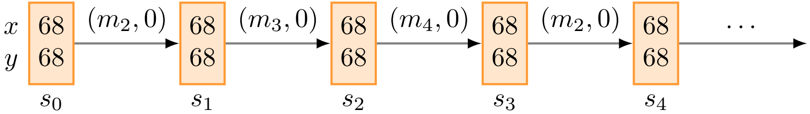


Keywords: State, Schedule, periodic schedule, ultimately periodic schedule, trajectory, and safe schedule

Multi-mode System: Zeno schedule



Safe set: $x \in [65, 70], y \in [65, 70]$



Keywords: Zeno Schedule

Multi-mode Systems: Zeno schedule

m_1: $\dot{x} = -2$, $\dot{y} = 3$

m_2: $\dot{x} = -1$, $\dot{y} = -1$

m_3: $\dot{x} = -1$, $\dot{y} = 3$

m_4: $\dot{x} = 2$, $\dot{y} = -2$

m_5: $\dot{x} = 2$, $\dot{y} = -1$

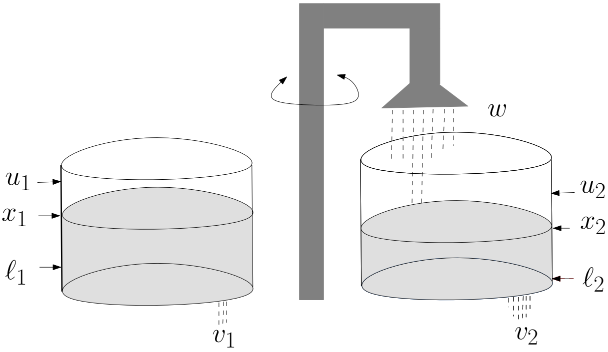
m_6: $\dot{x} = 2$, $\dot{y} = 3$

Safe set: $x \in [65, 70], y \in [65, 70]$

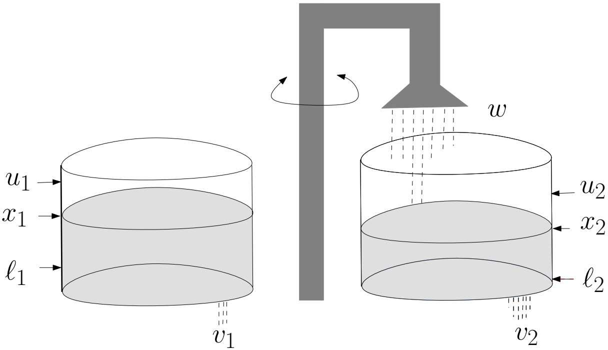
$$\begin{matrix} x \\ y \end{matrix}\ \underset{s_0}{\boxed{\begin{matrix} 68 \\ 68 \end{matrix}}} \xrightarrow{(m_2, 1)} \underset{s_1}{\boxed{\begin{matrix} 67 \\ 67 \end{matrix}}} \xrightarrow{(m_3, \frac{1}{2})} \underset{s_2}{\boxed{\begin{matrix} 66.5 \\ 68.5 \end{matrix}}} \xrightarrow{(m_4, \frac{1}{4})} \underset{s_3}{\boxed{\begin{matrix} 67 \\ 68 \end{matrix}}} \xrightarrow{(m_2, \frac{1}{8})} \underset{s_4}{\boxed{\begin{matrix} 66.875 \\ 67.875 \end{matrix}}} \xrightarrow{\cdots}$$

Keywords: Zeno Schedule

Another Example: Leaking Tanks Systems



Another Example: Leaking Tanks Systems



$$\dot{x_1} = -v_1 \qquad \dot{x_1} = w - v_1 \qquad \dot{x_1} = -v_1$$
$$\dot{x_2} = -v_1 \qquad \dot{x_2} = -v_2 \qquad \dot{x_2} = w - v_2$$

m_1 $\quad$ m_2 $\quad$ m_3

$$x_1 \in [\ell_1, u_1], x_2 \in [\ell_2, u_2]$$

… and more

1. Temperature and humidity control in cloud servers
2. Robot motion planning
3. Autonomous vehicles navigation
4. and more..

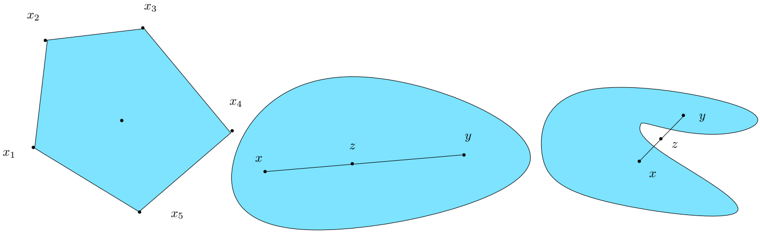
Motivation

Constant-Rate Multi-Mode Systems

Optimization, Discretization, and Undecidability

Summary

Definitions: Convex Sets

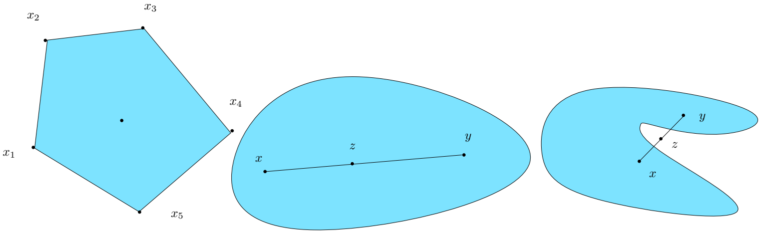


Convex Sets

Non-Convex Set

– A convex combination of a set of points $x_1, x_2, \ldots, x_n \in \mathbb{R}^n$ is a point of the form $\lambda_1 x_1 + \lambda_2 x_2 + \cdots + \lambda_n x_n$ where $\lambda_i \in [0, 1]$ and $\sum_i \lambda_i = 1$.

Definitions: Convex Sets

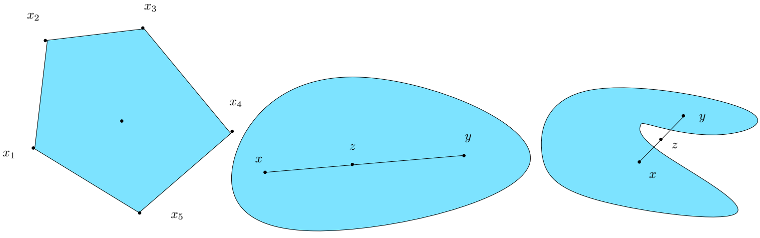


Convex Sets

Non-Convex Set

- A convex combination of a set of points $x_1, x_2, \ldots, x_n \in \mathbb{R}^n$ is a point of the form $\lambda_1 x_1 + \lambda_2 x_2 + \cdots + \lambda_n x_n$ where $\lambda_i \in [0, 1]$ and $\sum_i \lambda_i = 1$.
- A set $S \subseteq \mathbb{R}^n$ is convex if for any set of points $x_1, x_2, \ldots, x_n \in S$ their convex combinations are also in S.

Definitions: Convex Sets



Convex Sets

Non-Convex Set

- A convex combination of a set of points $x_1, x_2, \ldots, x_n \in \mathbb{R}^n$ is a point of the form $\lambda_1 x_1 + \lambda_2 x_2 + \cdots + \lambda_n x_n$ where $\lambda_i \in [0, 1]$ and $\sum_i \lambda_i = 1$.
- A set $S \subseteq \mathbb{R}^n$ is convex if for any set of points $x_1, x_2, \ldots, x_n \in S$ their convex combinations are also in S.
- The convex hull of points $x_1, x_2, \ldots, x_n \in \mathbb{R}^n$ is the minimum convex set that contains these point, and is the set of all convex combinations.

Formal Definitions

Definition (Constant-Rate Multi-Mode Systems: MMS)

A MMS is a tuple $\mathcal{H} = (M, n, R)$ where

- M is a finite nonempty set of modes,
- n is the number of continuous variables,
- $R : M \to \mathbb{R}^n$ gives for each mode the rate vector,
- $S \subseteq \mathbb{R}^n$ is a bounded convex set of safe states.

Formal Definitions

Definition (Constant-Rate Multi-Mode Systems: MMS)

A MMS is a tuple $\mathcal{H} = (M, n, R)$ where

- M is a finite nonempty set of modes,
- n is the number of continuous variables,
- $R : M \to \mathbb{R}^n$ gives for each mode the rate vector,
- $S \subseteq \mathbb{R}^n$ is a bounded convex set of safe states.

- The trajectory of a schedule $(m_1, t_1), (m_2, t_2), \ldots, (m_k, t_k)$ from s_0 is

$$s_0, (m_1, t_1), s_1, \ldots, (m_k, t_k), s_k$$

  such that $s_i = s_{i-1} + t_i \cdot R(m_i)$ for all for all $1 \leq i \leq k$.
- A schedule is safe at s_0 if all states of its trajectory from s_0 are safe.
- A mode m is t-safe at a state $s \in S$ if the schedule (m, t) is safe.

Definition

Safe Schedulability Problem

Given an MMS $\mathcal{H}$ and a starting state s_0 decide whether there exists a non-Zeno safe schedule.

Definition

Safe Schedulability Problem

Given an MMS $\mathcal{H}$ and a starting state s_0 decide whether there exists a non-Zeno safe schedule.

Theorem

Safe Schedulability can be solved in polynomial time.

Definition

Safe Schedulability Problem

Given an MMS $\mathcal{H}$ and a starting state s_0 decide whether there exists a non-Zeno safe schedule.

Theorem

Safe Schedulability can be solved in polynomial time.

Safe Reachability Problem

Given an MMS $\mathcal{H}$, a starting state $s_0 \in S$, and a target state $s_t \in S$, decide whether there exists a safe schedule that reaches s_t from s_0.

Definition

Safe Schedulability Problem

Given an MMS $\mathcal{H}$ and a starting state s_0 decide whether there exists a non-Zeno safe schedule.

Theorem

Safe Schedulability can be solved in polynomial time.

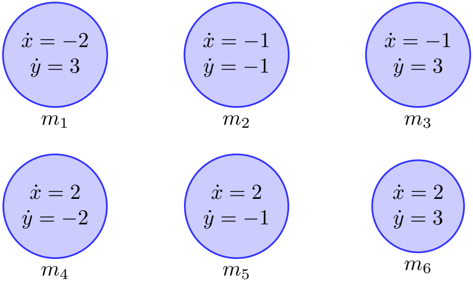
Safe Reachability Problem

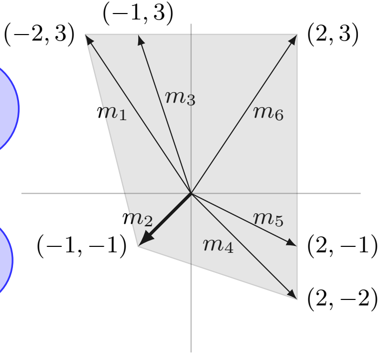
Given an MMS $\mathcal{H}$, a starting state $s_0 \in S$, and a target state $s_t \in S$, decide whether there exists a safe schedule that reaches s_t from s_0.

Theorem

Safe Reachability can be solved in polynomial time if the starting and the target states lie in the interior of S.

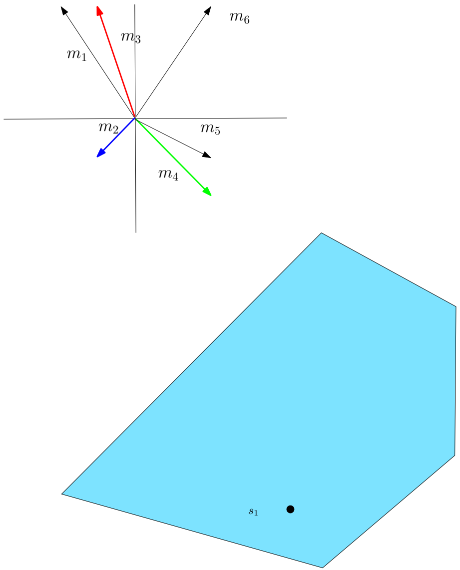
Safe Schedulability Problem: Geometry



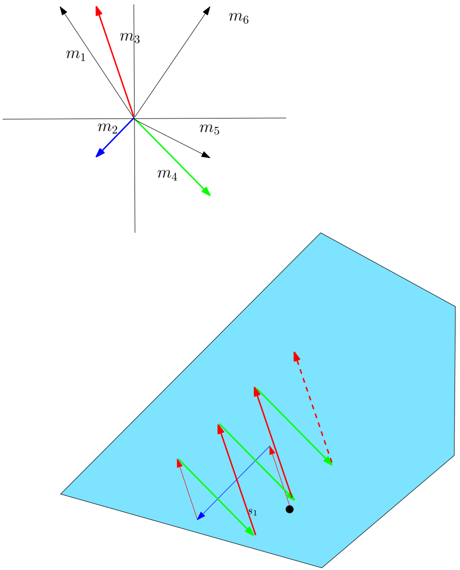


Safe set: $x \in [65, 70], y \in [65, 70]$

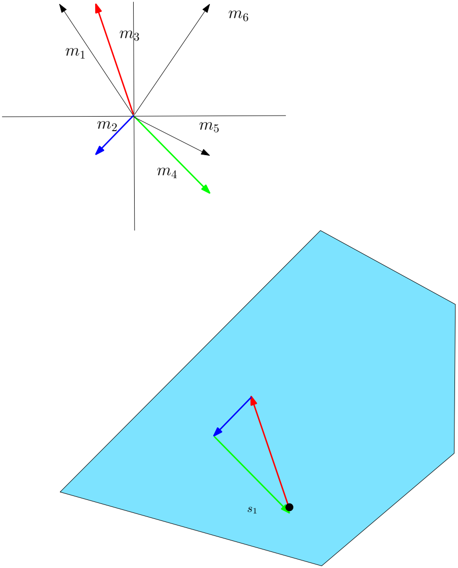
Safe Schedulability Problem: Geometry



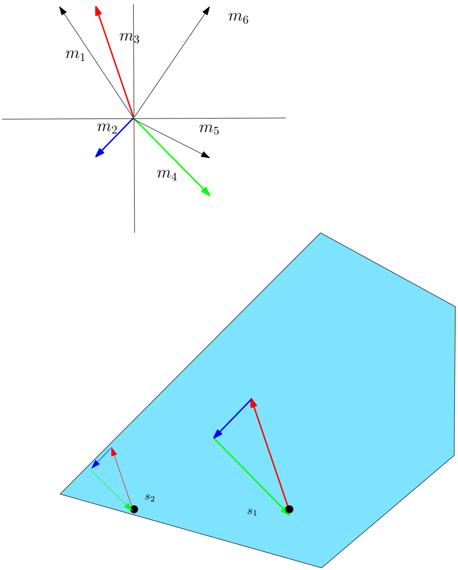
Safe Schedulability Problem: Geometry



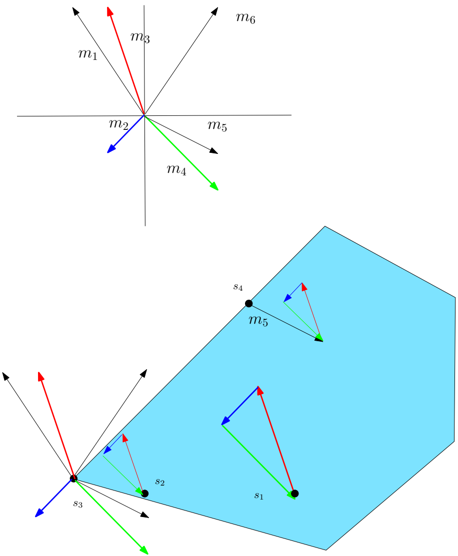
Safe Schedulability Problem: Geometry



Safe Schedulability Problem: Geometry



Safe Schedulability Problem: Geometry



Safe Schedulability Problem: Interior Case

Lemma

Assume that the starting state lies in the interior of the safety set. A safe non-Zeno schedule exists if and only if

$$\begin{aligned} \sum_{i=1}^{|M|} R(i) \cdot f_i &= 0 \\ \sum_{i=1}^{|M|} f_i &= 1. \end{aligned}$$

for some $f_1, f_2, \ldots, f_{|M|} \geq 0$.
Moreover, such a schedule is periodic.

Safe Schedulability Problem: Interior Case

Proof Sketch: ("if" direction):

If for some non-negative f_i we have

$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

then there exists a non-Zeno periodic safe schedule.

Safe Schedulability Problem: Interior Case

Proof Sketch: ("if" direction):

If for some non-negative f_i we have

$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

then there exists a non-Zeno periodic safe schedule.

1. There exists a $t > 0$ such that all modes are safe at s_0 for t-time.
2. Consider the periodic schedule

$$(m_1, t \cdot f_1), (m_2, t \cdot f_2), \ldots, (m_{|M|}, t \cdot f_{|M|})$$

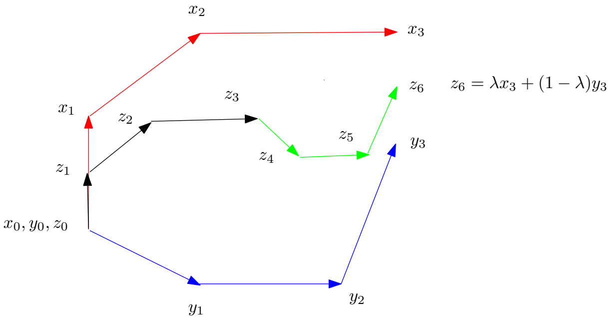
3. Notice that the schedule is non-Zeno.
4. Consider the trajectory of the schedule

$$s_0, (m_1, t_1), s_1(m_2, t_2), \ldots, s_{|M|}, (m_1, t_1) \ldots$$

5. Notice that $s_{i \cdot |M| + j} = s_j$ for all $i \geq 0$.
6. We show that $s_0, s_1, \ldots, s_{|M|-1}$ are safe.

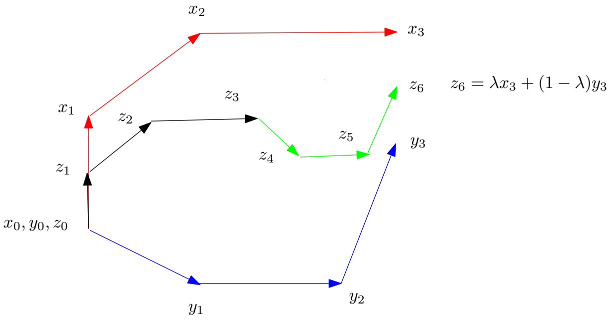
Safe Schedulability Problem: "If" Direction

Lemma: All convex combinations of finite safe schedules are safe.



Safe Schedulability Problem: “If” Direction

Lemma: All convex combinations of finite safe schedules are safe.



Corollary: All intermediate states visited in the following periodic schedule are safe if each mode is safe for time $t > 0$.

$$(m_1, t \cdot f_1), (m_2, t \cdot f_2), \ldots, (m_{|M|}, t \cdot f_{|M|})$$

Safe Schedulability Problem: Interior Case

Proof Sketch: ("only if" direction):

There exists a non-Zeno periodic safe schedule only if for some non-negative f_i we have

$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

Safe Schedulability Problem: Interior Case

Proof Sketch: ("only if" direction):

There exists a non-Zeno periodic safe schedule only if for some non-negative f_i we have

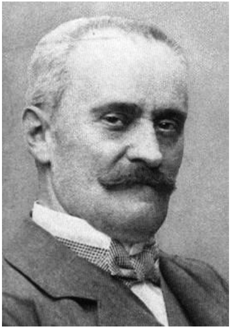
$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

1. Assume that it is not feasible.
2. Then by Farkas's lemma there is $(v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$ such that

$$(v_1, v_2, \ldots, v_n) \cdot R(i) > 0 \text{ for all modes } i.$$

3. Taking any mode contributes to some progress in the direction $(v_1, v_2, \ldots, v_n)$
4. Any non-Zeno schedule will eventually leave the safety set.

Farkas's Lemma



Gyula Farkas (1847–1930)

Theorem

Let A be a real $N \times M$ matrix and b be an N-dimensional vector. Then exactly one of the following two statements is true.

- *There exists a vector $x \in \mathbb{R}^M$ such that $Ax = b$ and $x \geq 0$.*
- *There exists a vector $y \in \mathbb{R}^N$ such that $A^T y \geq 0$ and $b^T y < 0$.*

Farkas's lemma application

There exists a vector $(f_1, f_2, \ldots, f_m)$ such that for

$$A = \begin{pmatrix} R(1)(1) & R(2)(1) & \cdots & R(m)(1) \\ R(1)(2) & R(2)(2) & \cdots & R(m)(2) \\ R(1)(3) & R(2)(3) & \cdots & R(m)(3) \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

$$x = (f_1, f_2, \ldots, f_m) \text{ and } b = (0, 0, \ldots, 1),$$

we have that $Ax = b$ and $x \geq 0$.

Farkas's lemma application

There exists a vector $(f_1, f_2, \ldots, f_m)$ such that for

$$A = \begin{pmatrix} R(1)(1) & R(2)(1) & \cdots & R(m)(1) \\ R(1)(2) & R(2)(2) & \cdots & R(m)(2) \\ R(1)(3) & R(2)(3) & \cdots & R(m)(3) \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

$$x = (f_1, f_2, \ldots, f_m) \text{ and } b = (0, 0, \ldots, 1),$$

we have that $Ax = b$ and $x \geq 0$.
By Farkas's lemma, either our equations are feasible or the following is feasible.
There exists a vector $(v_1, v_2, \ldots, v_n, d)$ such that for

$$A^T = \begin{pmatrix} R(1)(1) & R(1)(2) & R(1)(3) & 1 \\ R(2)(1) & R(2)(2) & R(2)(3) & 1 \\ R(3)(1) & R(3)(2) & R(3)(3) & 1 \\ \cdots & & & \\ R(m)(1) & R(m)(2) & R(m)(3) & 1 \end{pmatrix}$$

and $b^t = (0, 0, 0, \ldots, 1)^T$, $A^T y \geq 0$ and $b^T y < 0$.

Safe Schedulability Problem: Interior Case

Proof Sketch: ("only if" direction):

There exists a non-Zeno periodic safe schedule only if for some non-negative f_i we have

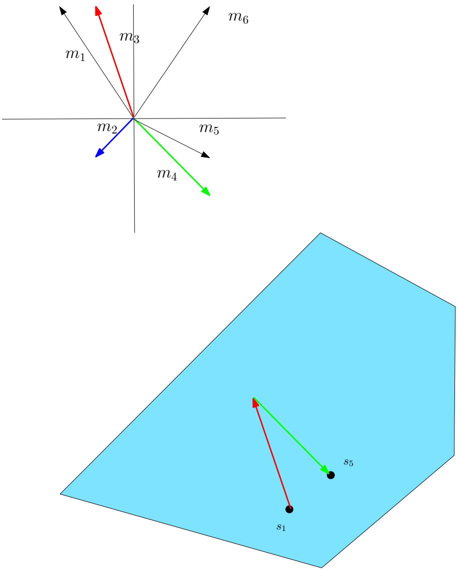
$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

1. Assume that it is not feasible.
2. Then by Farkas's lemma there is $(v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$ such that

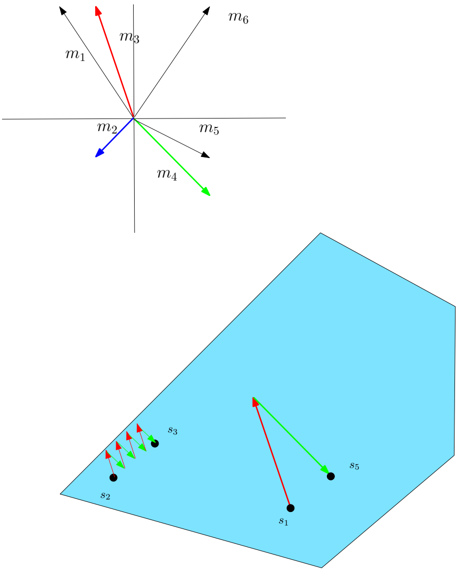
$$(v_1, v_2, \ldots, v_n) \cdot R(i) > 0 \text{ for all modes } i.$$

3. Taking any mode contributes to some progress in the direction $(v_1, v_2, \ldots, v_n)$
4. Any non-Zeno schedule will eventually leave the safety set.

Reachability Problem: Geometry



Reachability Problem: Geometry



Safe Reachability Problem

Lemma

Assume that the starting state s_0 and the target state s_t lie in the interior of the safety set.
A safe schedule exists from s_0 to s_t exists if and only if

$$s_0 + \sum_{i=1}^{|M|} R(i) \cdot t_i = s_t$$

for some $t_1, t_2, \ldots, t_{|M|} \geq 0$.

Proof Sketch:

"Only if" direction is trivial.

Safe Reachability Problem

Proof Sketch: ("if" direction):
If for some $t_1, t_2, \ldots, t_{|M|} \geq 0$ we have that

$$s_0 + \sum_{i=1}^{|M|} R(i) \cdot t_i = s_t$$

then a safe schedule exists from s_0 to s_t.

1. There exists a $t > 0$ such that all modes are safe at s_0 and s_t for t-time. Notice all points on the line connecting s_0 and s_t.
2. Let ℓ be a natural number greater than $\frac{\sum_{i=1}^{|M|} t_i}{t}$.
3. The periodic schedule $(m_1, t_1/\ell), (m_2, t_2/\ell), \ldots, (m_M, t_{|M|}/\ell)$ reaches the target in $\ell \cdot |M|$ steps.
4. Each intermediate state is in the safety set.

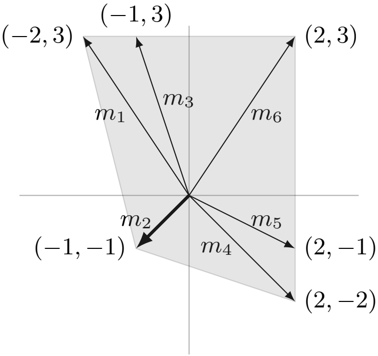
Thumb Rules: Schedulability

The following is feasible:

$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

Or, the following in infeasible:

$(v_1, v_2, \ldots, v_n) \cdot R(i) > 0$ for all modes i.



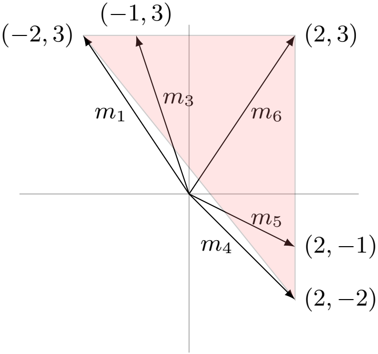
Thumb Rules: Schedulability

The following is feasible:

$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1$$

Or, the following in infeasible:

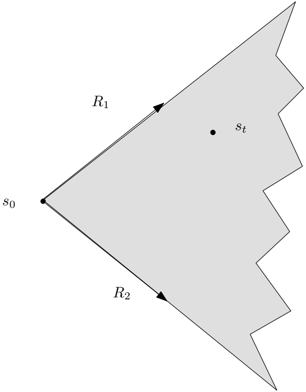
$(v_1, v_2, \ldots, v_n) \cdot R(i) > 0$ for all modes i.



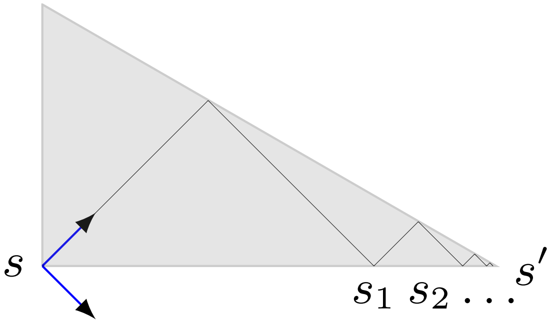
Thumb Rules: Reachability

The following is feasible:

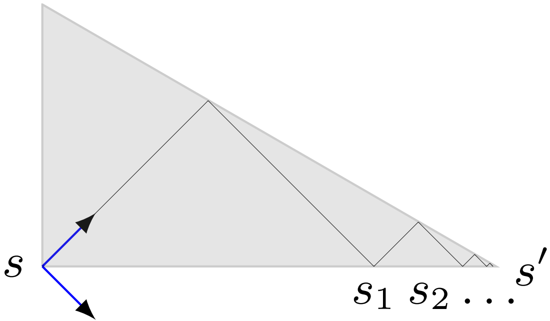
$$s_0 + \sum_{i=1}^{|M|} R(i) \cdot t_i = s_t$$



Reachability: Boundary Case

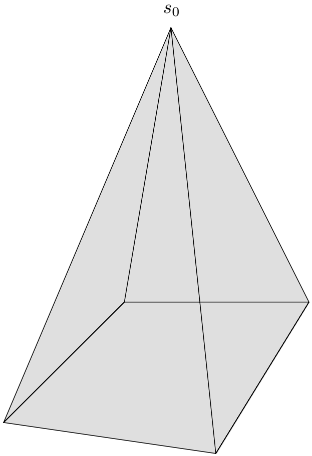


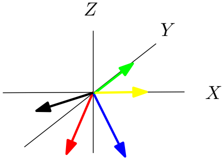
Reachability: Boundary Case



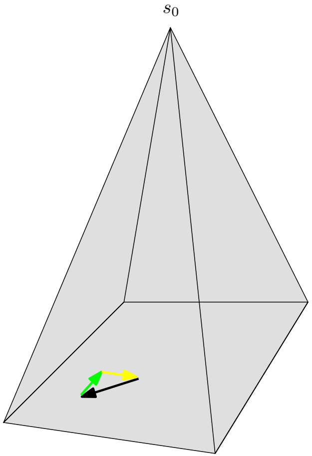
1. Rate vectors are $(1, 1)$ and $(1, -1)$
2. Angle at s' is 30^o.
3. $\|s_k, s\| = \|s, s'\| \cdot (\frac{\sqrt{3}-1}{\sqrt{3}+1})^k$.

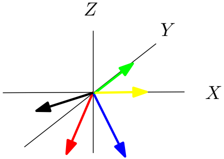
Schedulability: Boundary Case



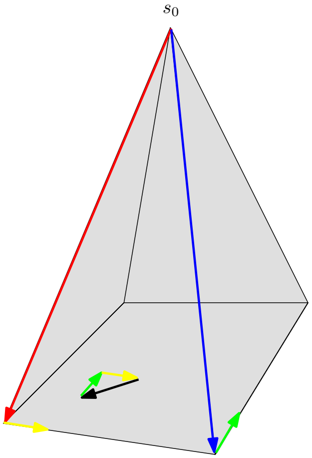


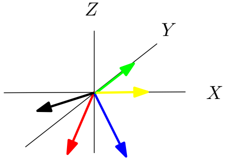
Schedulability: Boundary Case



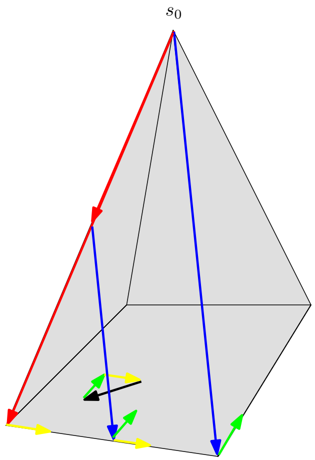


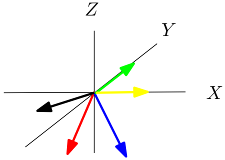
Schedulability: Boundary Case





Schedulability: Boundary Case





Schedulability: Boundary Case

Lemma

For any finite safe schedule σ there exists a finite safe schedule σ' s.t.:

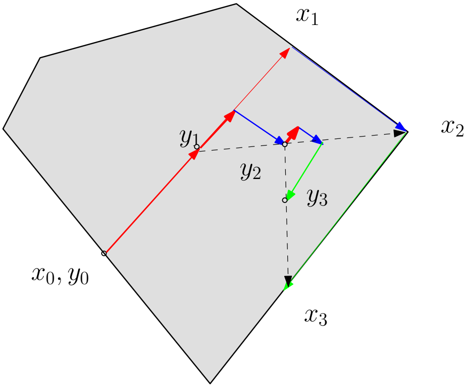
1. *All modes that were ever safe during the trajectory with σ will be safe in the final state of σ', and*
2. *The set of safe modes in every state of σ' will always be increasing.*

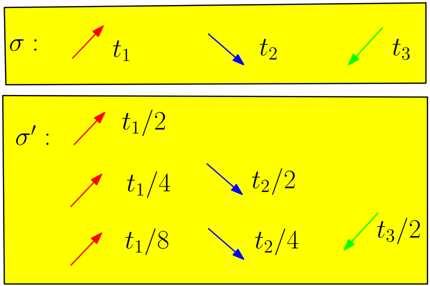
Schedulability: Boundary Case

Lemma

For any finite safe schedule σ there exists a finite safe schedule σ' s.t.:

1. *All modes that were ever safe during the trajectory with σ will be safe in the final state of σ', and*
2. *The set of safe modes in every state of σ' will always be increasing.*





Algorithm: Interior Case

1. Compute the sequence of set of modes $M_1, M_2, \ldots, M_k$ such that
   - M_1 is the set of safe modes at x_0, and
   - M_i is the set of safe modes at states reachable from x_0 using only modes from M_{i-1}.
2. $M_1 \subset M_2 \subset \cdots \subset M_k$.
3. Modes outside M_k are never reachable from x_0.
4. The set M_k can be computed in polynomial time.
5. MMS is schedulable from x_0 if and only if:

$$\sum_{m \in M_k} R(m) \cdot f_m = 0 \text{ and } \sum_{m \in M_k} f_m = 1$$

6. That can, again, be checked in polynomial time.

Motivation

Constant-Rate Multi-Mode Systems

Optimization, Discretization, and Undecidability

Summary

Optimization Schedulability and Reachability

- MMS $\mathcal{H} = (M, n, R)$ and price function $\pi : M \to \mathbb{R}$
- Price of a finite schedule $(m_1, t_1), (m_2, t_2), \ldots, (m_k, t_k)$ is

$$\sum_{i=1}^{k} \pi(m_i) t_i.$$

- Average price of an infinite schedule $(m_1, t_1), (m_2, t_2), \ldots$ is

$$\limsup_{n \to \infty} \frac{\sum_{i=1}^{k} \pi(m_i) t_i}{\sum_{i=1}^{k} t_i}.$$

- Optimal reachability-price and average-price problems

Optimization Schedulability and Reachability

- MMS $\mathcal{H} = (M, n, R)$ and price function $\pi : M \to \mathbb{R}$
- Price of a finite schedule $(m_1, t_1), (m_2, t_2), \ldots, (m_k, t_k)$ is

$$\sum_{i=1}^{k} \pi(m_i) t_i.$$

- Average price of an infinite schedule $(m_1, t_1), (m_2, t_2), \ldots$ is

$$\limsup_{n \to \infty} \frac{\sum_{i=1}^{k} \pi(m_i) t_i}{\sum_{i=1}^{k} t_i}.$$

- Optimal reachability-price and average-price problems
- Minimize $\sum_{i=1}^{|M|} t_i \cdot \pi(m_i)$ subject to:

$$s_0 + \sum_{i=1}^{|M|} R(i) \cdot t_i = s_t, \text{ and } t_i \geq 0.$$

- Minimize $\sum_{i=1}^{|M|} f_i \cdot \pi(m_i)$ subject to:

$$\sum_{i=1}^{|M|} R(i) \cdot f_i = 0 \text{ and } \sum_{i=1}^{|M|} f_i = 1, f_i \geq 0.$$

Discrete Schedulability and Undecidability

Discrete Schedulability:

- Requiring schedules with delays that are multiples of a given sampling rate
- For a bounded safety set only a finite number of states reachable using such discrete schedulers.
- Such reachable state-transition graph is of exponential size.
- schedulability/optimization problems can be solved in PSPACE.
- PSPACE-hardness can be shown by a reduction from the acceptance problem for linear-bounded automata.

Discrete Schedulability and Undecidability

Discrete Schedulability:

- Requiring schedules with delays that are multiples of a given sampling rate
- For a bounded safety set only a finite number of states reachable using such discrete schedulers.
- Such reachable state-transition graph is of exponential size.
- schedulability/optimization problems can be solved in PSPACE.
- PSPACE-hardness can be shown by a reduction from the acceptance problem for linear-bounded automata.

Generalizations:

- One can add some structure to the system by adding
  - guards on mode-switches
  - mode-dependent invariants
- Corresponds to a variant (singular) of hybrid automata [HKPV98]
- Each of these generalizations lead to undecidability of the reachability problem.

Undecidability Proof

Marvin Minsky (1927)

A Minsky machine $\mathcal{A}$ is a tuple (L, C, D) where:

- $L = \{\ell_0, \ell_1, \ldots, \ell_n\}$ is the set of states including the distinguished terminal state ℓ_n;
- $C = \{c_1, c_2\}$ is the set of two counters;
- $D = \{\delta_0, \delta_1, \ldots, \delta_{n-1}\}$ is the set of transitions of the following type:
  1. $c := c + 1$; goto ℓ_k,
  2. if $(c > 0)$ then $(c := c - 1;$ goto $\ell_k)$ else goto ℓ_m,

Undecidability Proof

- Let $\mathcal{A} = (L, C, D)$ be a Minsky machine.
- A configuration of a Minsky machine is a tuple (ℓ, c, d)
- The initial configuration $(\ell_0, 0, 0)$
- The run of a Minsky machine is a (finite or infinite) valid sequence of configurations $\langle k_0, k_1, \ldots \rangle$
- The run is a finite sequence (halting) if and only if the last configuration is the terminal state ℓ_n.
- The halting problem for a Minsky machine asks whether its unique run is finite.

Theorem ([Min67])

The halting problem for the two-counter Minsky machines is undecidable.

Undecidability Proof

- Let $\mathcal{A} = (L, C, D)$ be a Minsky machine.
- A configuration of a Minsky machine is a tuple (ℓ, c, d)
- The initial configuration $(\ell_0, 0, 0)$
- The run of a Minsky machine is a (finite or infinite) valid sequence of configurations $\langle k_0, k_1, \ldots \rangle$
- The run is a finite sequence (halting) if and only if the last configuration is the terminal state ℓ_n.
- The halting problem for a Minsky machine asks whether its unique run is finite.

Theorem ([Min67])

The halting problem for the two-counter Minsky machines is undecidable.

We reduce Minsky machine halting problem to singular hybrid automata reachability problem.

Undecidability Proof

- Let $\mathcal{A} = (L, C, D)$ be a Minsky machine.
- A configuration of a Minsky machine is a tuple (ℓ, c, d)
- The initial configuration $(\ell_0, 0, 0)$
- The run of a Minsky machine is a (finite or infinite) valid sequence of configurations $\langle k_0, k_1, \ldots \rangle$
- The run is a finite sequence (halting) if and only if the last configuration is the terminal state ℓ_n.
- The halting problem for a Minsky machine asks whether its unique run is finite.

Theorem ([Min67])

The halting problem for the two-counter Minsky machines is undecidable.

We reduce Minsky machine halting problem to singular hybrid automata reachability problem.

Theorem

The reachability problem for singular hybrid automata is undecidable.

Motivation

Constant-Rate Multi-Mode Systems

Optimization, Discretization, and Undecidability

Summary

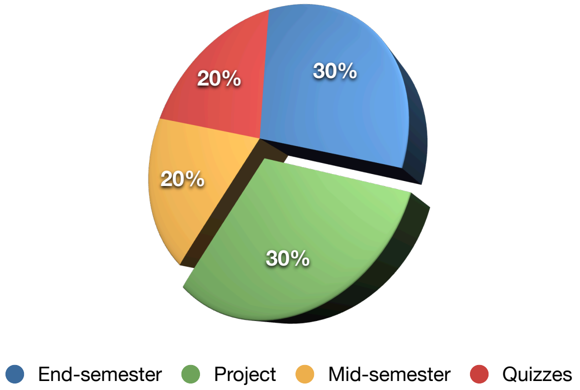
Summary

1. Discussed a model for constant-rate multi-mode systems
2. Polynomial-time algorithms for safe schedulability and safe reachability
3. Energy peak demand reduction problem
4. Discrete schedulers lead to PSPACE-hardness
5. Adding either local invariants or guards lead to undecidability
6. Bounded-rate Multi-mode systems

Course overview

1. Formal Modeling of CPS
   - Discrete Dynamical Systems (Extended Finite State Machines)
   - Continuous Dynamical Systems (Ordinary Differential Equations)
   - Hybrid Dynamical Systems
     - Timed automata,
     - Hybrid automata,
     - PCDs, Multi-mode systems, and other decidable subclasses
2. Tools for modeling CPS
   - UPPAAL
   - HyTech
   - Stateflow/Simulink
3. Verification and Synthesis
   - Classical temporal logics LTL and CTL
   - Real-time extensions of these logics, in particular MTL
   - Model-Checking for timed and hybrid automata
   - Automatic Synthesis for CPS (satisfiability, controller-environment games, code-generations, etc.)

Grading



T. A. Henzinger, P. W. Kopke, A. Puri, and P. Varaiya.
What's decidable about hybrid automata?
Journal of Comp. and Sys. Sciences, 57:94–124, 1998.

Marvin L. Minsky.
Computation: finite and infinite machines.
Prentice-Hall, Inc., 1967.

T. X. Nghiem, M. Behl, G. J. Pappas, and R. Mangharam.
Green scheduling: Scheduling of control systems for peak power reduction.
2nd International Green Computing Conference, July 2011.