Reachability in vector addition systems

Kosaraju's proof, exposited in *"The Mathematics of Petri Nets"* by C. Reutenauer (translated by I. Craig)

Kamal Lodaya and M. Praveen

The Institute of Mathematical Sciences, Chennai

Formal Methods Update Meeting, IIT Roorkee, July 2009

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

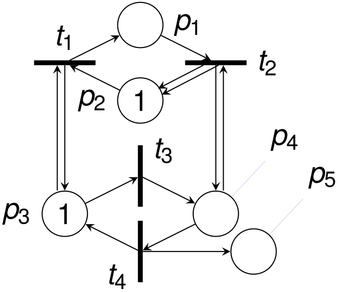


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

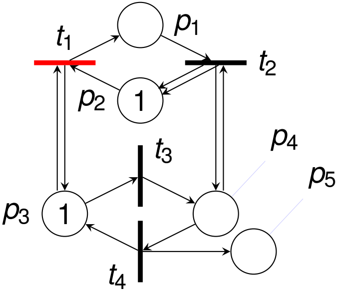


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

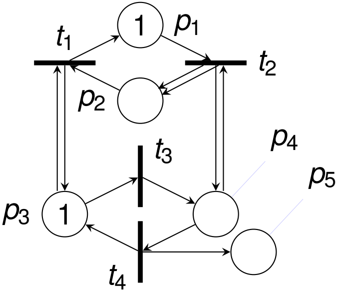


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

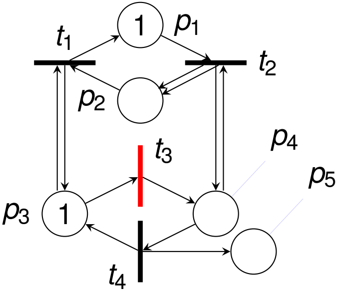


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

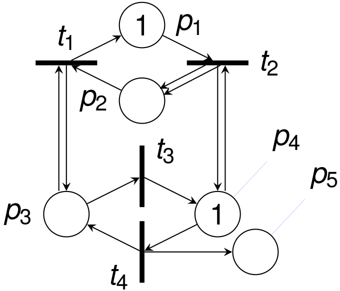


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

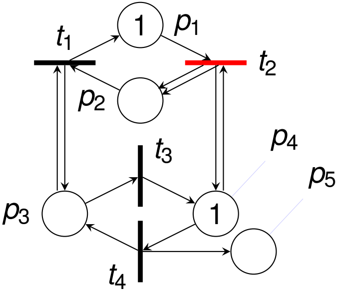


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

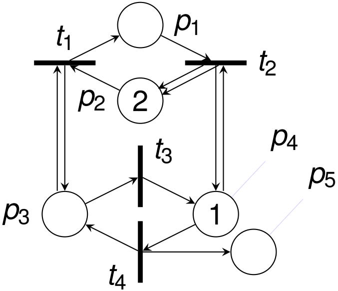


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

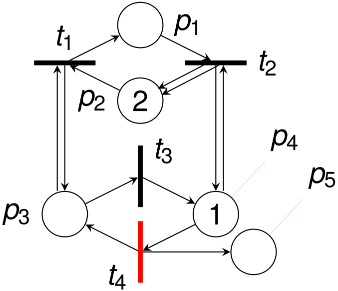


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

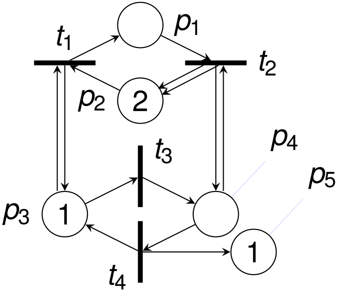


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

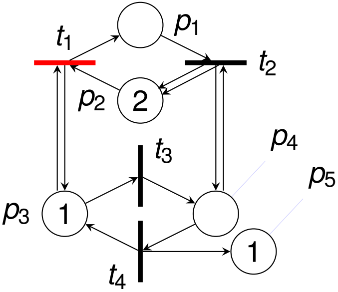


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

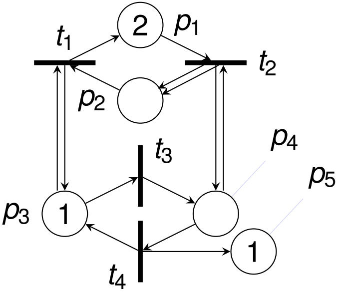


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

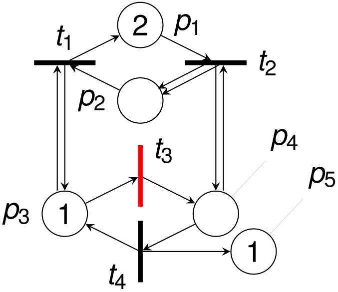


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

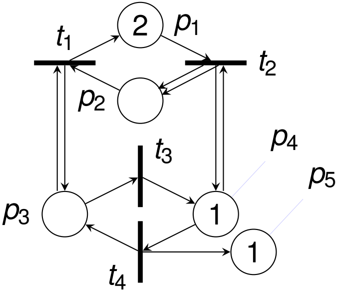


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

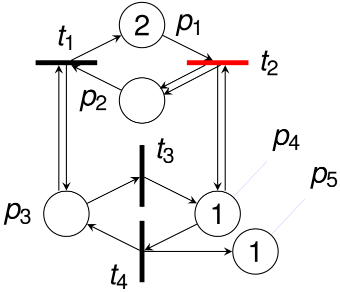


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

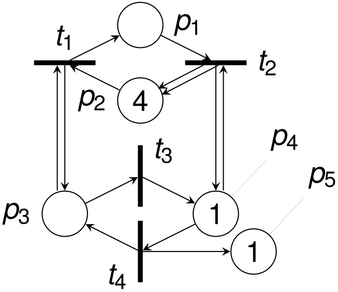


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

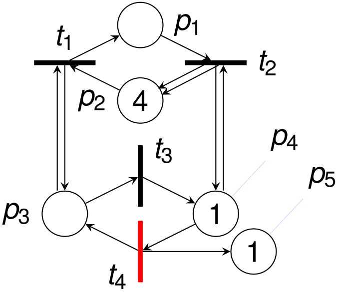


Figure: Hopcroft and Pansiot's example Petri net

Petri nets - Introduction

- Mathematical model.
- Widely used to study systems with concurrent processes.

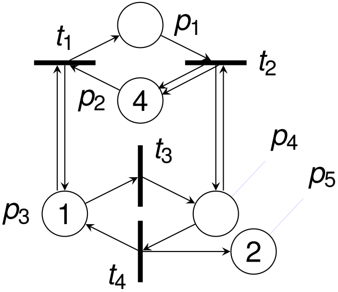
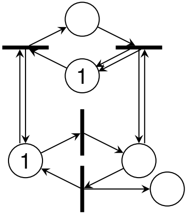


Figure: Hopcroft and Pansiot's example Petri net

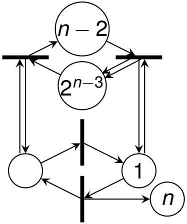
Reachability problem

Starting from



$$M_i = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

, can we reach



$$M_f = \begin{bmatrix} n-2 \\ 2^{n-3} \\ 0 \\ 1 \\ n \end{bmatrix}$$

?

Work on decidability of the reachability problem

- E.W.Mayr gave an algorithm for the general Petri net reachability problem in 1981/1984.
- S.R.Kosaraju and J.L.Lambert simplified the proofs in 1982 and 1992.
- No upper bound known for the above algorithm. In the worst case, it requires more than primitive recursive space.
- R.J.Lipton gave an exponential space lower bound for the general Petri net reachability problem.
- J. Leroux has published a new algorithm that uses a different approach, but proof of correctness depends on ideas from the earlier algorithm.
- K. Reinhardt extended the idea to decide reachability in Petri nets where inhibitor arcs occur in a restricted way.

A naive approach - reachability graph

Start with the initial marking and grow a tree of reachable markings.

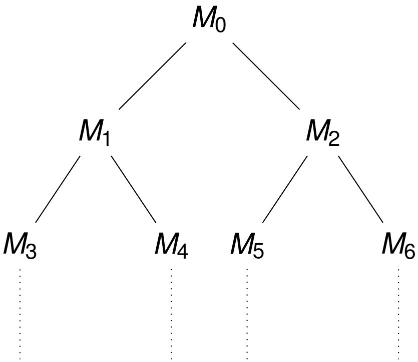


Figure: Reachability graph

Another naive approach - incidence matrix

$$\mathbf{N} = \begin{array}{c} \\ p_1 \\ p_2 \\ \vdots \\ p_m \end{array} \left[\begin{array}{rccc} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{array} \right]$$

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

- If $M_0 \xrightarrow{\sigma} M_f$, the Parikh vector $\overline{\sigma}$ will satisfy the above equation.

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

- If $M_0 \xrightarrow{\sigma} M_f$, the Parikh vector $\overline{\sigma}$ will satisfy the above equation.
- The converse need not be true.

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

- If $M_0 \xrightarrow{\sigma} M_f$, the Parikh vector $\overline{\sigma}$ will satisfy the above equation.
- The converse need not be true.
- Try all solutions.

Another naive approach - incidence matrix

- State equation

$$\begin{bmatrix} M_0(p_1) \\ M_0(p_2) \\ \vdots \\ M_0(p_m) \end{bmatrix} + \begin{bmatrix} t_1 & t_2 & \cdots & t_n \\ -1 & & & \\ +2 & & & \\ & & & \\ 0 & & & \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} M_f(p_1) \\ M_f(p_2) \\ \vdots \\ M_f(p_m) \end{bmatrix}$$

- If $M_0 \xrightarrow{\sigma} M_f$, the Parikh vector $\overline{\sigma}$ will satisfy the above equation.
- The converse need not be true.
- Try all solutions.
- If vector $\mathbf{I}$ is such that $\mathbf{N} \times \mathbf{I} = \mathbf{0}$, there will be infinitely many solutions.
- If $\mathbf{N} \times \mathbf{I} = \mathbf{0}$, $\mathbf{I}$ is called a T-invariant.

Idea of the algorithm

- If something is finite, hold on to it!

Idea of the algorithm

- If something is finite, hold on to it!
- All solutions to the state equation $M_0 + \mathbf{N}\mathbf{X} = M_f$ are contained in $\mathbf{B} + \mathbf{J}^*$, where
  - $\mathbf{B} = \{B_1, \ldots, B_r\}$ is the finite set of *minimal solutions*.
  - $\mathbf{J} = \{I_1, \ldots, I_s\}$ is a finite set of T-invariants, that generates all invariants.

Idea of the algorithm

- If something is finite, hold on to it!
- All solutions to the state equation $M_0 + \mathbf{NX} = M_f$ are contained in $\mathbf{B} + \mathbf{J}^*$, where
  - $\mathbf{B} = \{B_1, \ldots, B_r\}$ is the finite set of *minimal solutions*.
  - $\mathbf{J} = \{I_1, \ldots, I_s\}$ is a finite set of T-invariants, that generates all invariants.
- If the co-ordinate corresponding to a transition t is 0 in all the vectors $I_1, \ldots, I_s$, then it is not part of any T-invariant.
- t may be used at most w times, determined by $B_1, \ldots, B_r$.

Idea of the algorithm

- If something is finite, hold on to it!
- All solutions to the state equation $M_0 + \mathbf{NX} = M_f$ are contained in $\mathbf{B} + \mathbf{J}^*$, where
  - $\mathbf{B} = \{B_1, \ldots, B_r\}$ is the finite set of *minimal solutions*.
  - $\mathbf{J} = \{I_1, \ldots, I_s\}$ is a finite set of T-invariants, that generates all invariants.
- If the co-ordinate corresponding to a transition t is 0 in all the vectors $I_1, \ldots, I_s$, then it is not part of any T-invariant.
- t may be used at most w times, determined by $B_1, \ldots, B_r$.
- Create w new Petri nets $\mathcal{N}_1, \ldots, \mathcal{N}_w$, where $\mathcal{N}_i$ allows t to be fired exactly i times.

Using a transition boundedly many times

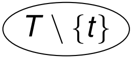


Figure: A chain of Vector Addition System with States

Using a transition boundedly many times

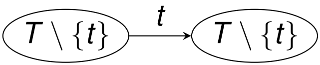


Figure: A chain of Vector Addition System with States

Using a transition boundedly many times

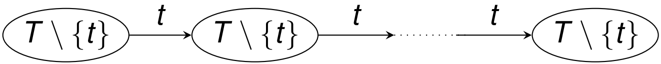


Figure: A chain of Vector Addition System with States

Using a transition boundedly many times

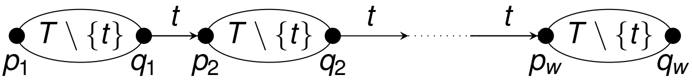


Figure: A chain of Vector Addition System with States

- p_i are entry states and q_i are exit states.

Using a transition boundedly many times

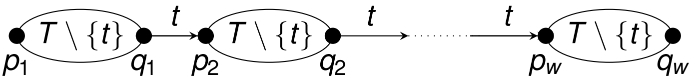


Figure: A chain of Vector Addition System with States

- p_i are entry states and q_i are exit states.
- With each transition t is associated a vector $effect(t)$ that denotes its effect on the places of the Petri net.

Using a transition boundedly many times

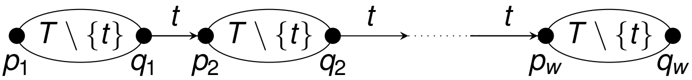


Figure: A chain of Vector Addition System with States

- p_i are entry states and q_i are exit states.
- With each transition t is associated a vector *effect*(t) that denotes its effect on the places of the Petri net.
- We need to check if starting from (p_1, M_0), we can reach (q_w, M_f). This is a chain of Constrained Vector Addition System with States (CVASS chain).

Using a transition boundedly many times

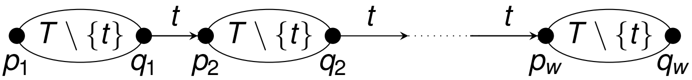


Figure: A chain of Vector Addition System with States

- p_i are entry states and q_i are exit states.
- With each transition t is associated a vector $\mathit{effect}(t)$ that denotes its effect on the places of the Petri net.
- We need to check if starting from (p_1, M_0), we can reach (q_w, M_f). This is a chain of Constrained Vector Addition System with States (CVASS chain).
- Number of transitions in each CVASS of the chain is strictly less than the number of transitions in the original CVASS.

Calculating bound on transitions - another way

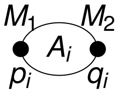


Figure: A constrained CVASS

- Consider the regular language $L \subseteq A_i^*$ consisting of paths from p_i to q_i (ignore the effect on the vector).

Calculating bound on transitions - another way

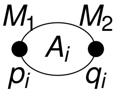


Figure: A constrained CVASS

- Consider the regular language $L \subseteq A_i^*$ consisting of paths from p_i to q_i (ignore the effect on the vector).
- The set of Parikh images of strings in L will be of the form $\overline{L} = \mathbf{B} + \mathbf{J}^*$.

Calculating bound on transitions - another way

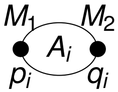


Figure: A constrained CVASS

- Consider the regular language $L \subseteq A_i^*$ consisting of paths from p_i to q_i (ignore the effect on the vector).
- The set of Parikh images of strings in L will be of the form $\overline{L} = \mathbf{B} + \mathbf{J}^*$.
- A set of vectors of the form $B_1 + \mathbf{J}^*$ is called a linear set. Finite union of linear sets is a semilinear set. Vectors in $\mathbf{J}$ are called periods.

Calculating bound on transitions - another way

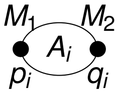


Figure: A constrained CVASS

- Consider the regular language $L \subseteq A_i^*$ consisting of paths from p_i to q_i (ignore the effect on the vector).
- The set of Parikh images of strings in L will be of the form $\overline{L} = \mathbf{B} + \mathbf{J}^*$.
- A set of vectors of the form $B_1 + \mathbf{J}^*$ is called a linear set. Finite union of linear sets is a semilinear set. Vectors in $\mathbf{J}$ are called periods.
- We need to handle entry and exit constraints also.

Calculating bound on transitions - Contd. . .

- Suppose there are m places to be handled by the vector and n transitions.
- For a string $\sigma \in A_i^*$, $(\overline{\sigma}, \mathit{effect}(\sigma))$ is a vector in $\mathbb{Z}^{n+m}$. First n co-ordinates is the Parikh image of σ and last m co-ordinates gives the change induced by σ on the places. This is the extended commutative image $\mathit{eci}(\sigma)$.

Calculating bound on transitions - Contd...

- Suppose there are m places to be handled by the vector and n transitions.
- For a string $\sigma \in A_i^*$, $(\overline{\sigma}, \mathit{effect}(\sigma))$ is a vector in $\mathbb{Z}^{n+m}$. First n co-ordinates is the Parikh image of σ and last m co-ordinates gives the change induced by σ on the places. This is the extended commutative image $\mathit{eci}(\sigma)$.
- $(0^n, M_1) + \mathit{eci}(\sigma)$ is a vector, which gives
  - Parikh image of σ in the first n co-ordinates.
  - Final vector reached if σ is fired from M_1, in the last m co-ordinates.

Calculating bound on transitions - Contd. . .

- Suppose there are m places to be handled by the vector and n transitions.
- For a string $\sigma \in A_i^*$, $(\overline{\sigma}, \mathit{effect}(\sigma))$ is a vector in $\mathbb{Z}^{n+m}$. First n co-ordinates is the Parikh image of σ and last m co-ordinates gives the change induced by σ on the places. This is the extended commutative image $\mathit{eci}(\sigma)$.
- $(0^n, M_1) + \mathit{eci}(\sigma)$ is a vector, which gives
  - Parikh image of σ in the first n co-ordinates.
  - Final vector reached if σ is fired from M_1, in the last m co-ordinates.
- $(0^n, M_1) + \mathit{eci}(L)$ is a semilinear set. Intersect it with the set of vectors $(\mathbb{N}^n, M_2)$. We will get another semilinear set that represents Parikh images of paths from p_i to q_i that satisfy the constraint.

Calculating bound on transitions - Contd. . .

- Suppose there are m places to be handled by the vector and n transitions.
- For a string $\sigma \in A_i^*$, $(\overline{\sigma}, effect(\sigma))$ is a vector in $\mathbb{Z}^{n+m}$. First n co-ordinates is the Parikh image of σ and last m co-ordinates gives the change induced by σ on the places. This is the extended commutative image $eci(\sigma)$.
- $(0^n, M_1) + eci(\sigma)$ is a vector, which gives
  - Parikh image of σ in the first n co-ordinates.
  - Final vector reached if σ is fired from M_1, in the last m co-ordinates.
- $(0^n, M_1) + eci(L)$ is a semilinear set. Intersect it with the set of vectors $(\mathbb{N}^n, M_2)$. We will get another semilinear set that represents Parikh images of paths from p_i to q_i that satisfy the constraint.
- If the co-ordinate corresponding to a transition t is 0 in all the periods of the above semilinear set, t can be used only boundedly many times.

Constraints at intermediate entry/exit states

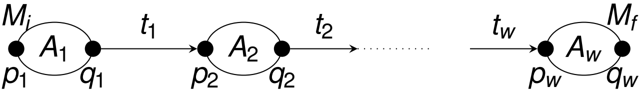


Figure: A chain of Vector Addition System with States

- L: language of strings from p_1 to q_w. For $\sigma \in L$, $project[A_1 \cup \{t_1\} \cup \cdots \cup A_i](\sigma)$ gives the portion of σ up to q_i.

Constraints at intermediate entry/exit states

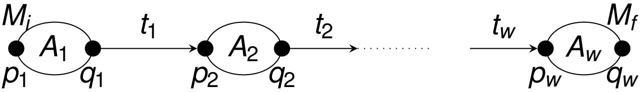


Figure: A chain of Vector Addition System with States

- L: language of strings from p_1 to q_w. For $\sigma \in L$, $project[A_1 \cup \{t_1\} \cup \cdots \cup A_i](\sigma)$ gives the portion of σ up to q_i.
- $effect(project[i](\sigma))$ gives the effect at q_i of firing σ at p_1.

Constraints at intermediate entry/exit states

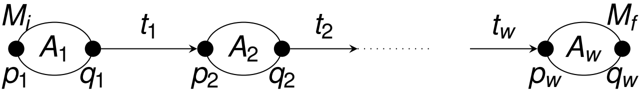


Figure: A chain of Vector Addition System with States

- L: language of strings from p_1 to q_w. For $\sigma \in L$, $project[A_1 \cup \{t_1\} \cup \cdots \cup A_i](\sigma)$ gives the portion of σ up to q_i.
- $effect(project[i](\sigma))$ gives the effect at q_i of firing σ at p_1.
- $(M_i + effect[i](\sigma), M_i + effect(\sigma))$ is a vector in $\mathbb{Z}^{2m}$ — first m co-ordinates give the result at q_i and last m co-ordinates give the result at q_w.

Constraints at intermediate entry/exit states

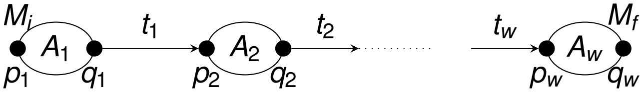


Figure: A chain of Vector Addition System with States

- L: language of strings from p_1 to q_w. For $\sigma \in L$, $project[A_1 \cup \{t_1\} \cup \cdots \cup A_i](\sigma)$ gives the portion of σ up to q_i.
- $effect(project[i](\sigma))$ gives the effect at q_i of firing σ at p_1.
- $(M_i + effect[i](\sigma), M_i + effect(\sigma))$ is a vector in $\mathbb{Z}^{2m}$ — first m co-ordinates give the result at q_i and last m co-ordinates give the result at q_w.
- $(M_i + effect[i], M_i + effect)(L)$ is a semilinear set. Intersect it with $(\mathbb{N}^m, M_f)$. Result is a semilinear set, whose vectors contain possible results at q_i while walking from (p_1, M_i) to (q_w, M_f).

Entry/exit constraints - Contd. . .

- If in the above semilinear set, the entry corresponding to a co-ordinate $j, 1 \leq j \leq m$ is 0 in all periods, that co-ordinate will never go beyond some bound given by the semilinear set.

Entry/exit constraints - Contd. . .

- If in the above semilinear set, the entry corresponding to a co-ordinate $j, 1 \leq j \leq m$ is 0 in all periods, that co-ordinate will never go beyond some bound given by the semilinear set.
- Such a co-ordinate is said to be constrained at the exit of i^{th} CVASS, with a bound say w.

Entry/exit constraints - Contd...

- If in the above semilinear set, the entry corresponding to a co-ordinate $j, 1 \leq j \leq m$ is 0 in all periods, that co-ordinate will never go beyond some bound given by the semilinear set.
- Such a co-ordinate is said to be constrained at the exit of i^{th} CVASS, with a bound say w.
- Create w new CVASS chains $\mathcal{N}_1, \ldots, \mathcal{N}_w$, where $\mathcal{N}_k$ puts k as a constraint in the co-ordinate j at q_i.

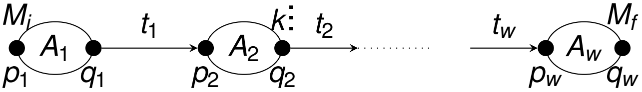


Figure: A chain of Constrained Vector Addition System with States

Entry/exit constraints - Contd. . .

- If in the above semilinear set, the entry corresponding to a co-ordinate $j, 1 \leq j \leq m$ is 0 in all periods, that co-ordinate will never go beyond some bound given by the semilinear set.
- Such a co-ordinate is said to be constrained at the exit of i^{th} CVASS, with a bound say w.
- Create w new CVASS chains $\mathcal{N}_1, \ldots, \mathcal{N}_w$, where $\mathcal{N}_k$ puts k as a constraint in the co-ordinate j at q_i.

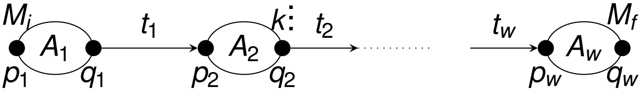


Figure: A chain of Constrained Vector Addition System with States

- In each of the w new CVASS chains, number of unconstrained co-ordinates at exit of i^{th} CVASS has decreased.

Constrained co-ordinates that are bounded

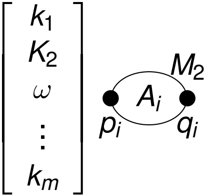


Figure: A constrained CVASS

- We want to find if within i^{th} CVASS, a co-ordinate can be bounded.

Constrained co-ordinates that are bounded

$$\begin{bmatrix} k_1 \\ K_2 \\ \omega \\ \vdots \\ k_m \end{bmatrix}$$

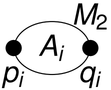


Figure: A constrained CVASS

- We want to find if within i^{th} CVASS, a co-ordinate can be bounded.
- Suppose the following sequence of transitions can be obtained: $\begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}, \ldots, \begin{bmatrix} 5 \\ 3 \\ 3 \end{bmatrix}, \ldots, \begin{bmatrix} 6 \\ 3 \\ 3 \end{bmatrix}$, where $(5, 3, 3)$ and $(6, 3, 3)$ are in the same state.

Bounded co-ordinates - Contd. . .

- A co-ordinate is unbounded iff there is such a “self covering” sequence. Existence of such sequences is decidable.

Bounded co-ordinates - Contd. . .

- A co-ordinate is unbounded iff there is such a "self covering" sequence. Existence of such sequences is decidable.
- If we find that a co-ordinate is bounded by say b, we will "get rid" of that co-ordinate and track its changes through states instead.
  If the first co-ordinate is bounded by 100 and the set of states in i^{th} CVASS is S, the new set of states will be $S \times \{0, \ldots, 100\}$. If $p \xrightarrow{t} q$, $effect(t) = (-1, \ldots, 2)$, it will be replaced by $(p, k+1) \xrightarrow{t'} (q, k)$, $effect(t') = (0, \ldots, 2)$.

Bounded co-ordinates - Contd. . .

- A co-ordinate is unbounded iff there is such a "self covering" sequence. Existence of such sequences is decidable.
- If we find that a co-ordinate is bounded by say b, we will "get rid" of that co-ordinate and track its changes through states instead.
  If the first co-ordinate is bounded by 100 and the set of states in i^{th} CVASS is S, the new set of states will be $S \times \{0, \ldots, 100\}$. If $p \xrightarrow{t} q$, $effect(t) = (-1, \ldots, 2)$, it will be replaced by $(p, k+1) \xrightarrow{t'} (q, k)$, $effect(t') = (0, \ldots, 2)$.
- If while exiting at q_i, value of the bounded co-ordinate is to be k, we will make (q_i, k) as the exit state.

Bounded co-ordinates - Contd. . .

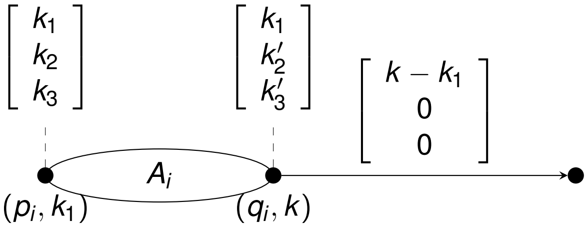


Figure: Bounded co-ordinates

The number of non-rigid co-ordinates has reduced in the i^{th} CVASS.

Reverse bounded co-ordinates

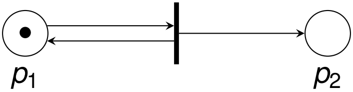


Figure: An unbounded Petri net

- Starting from $(1, 0)$, can we reach $(1, 50)$?

Reverse bounded co-ordinates

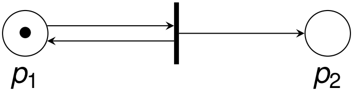


Figure: An unbounded Petri net

- Starting from $(1, 0)$, can we reach $(1, 50)$?
- p_2 is unbounded. Once we reach $(1, 51)$, can we go back to $(1, 50)$?

Reverse bounded co-ordinates

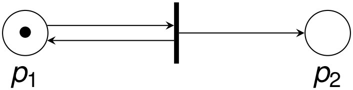


Figure: An unbounded Petri net

- Starting from $(1, 0)$, can we reach $(1, 50)$?
- p_2 is unbounded. Once we reach $(1, 51)$, can we go back to $(1, 50)$?
- Reverse the arcs, let the original final marking to be reached be the new initial marking and check for boundedness.

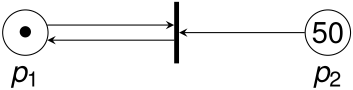


Figure: The reversed Petri net

Reverse bounded co-ordinates - Contd…

- In a CVASS, this amounts to reversing the arrows and making exit constraints as the new entry constraints.
- Just like an unbounded co-ordinate is due to a self covering sequence that pumps up the value, a reverse unbounded co-ordinate is due to a “self destroying” sequence that pumps down the value.

Will it ever stop? — Size of a CVASS chain

- The size of a CVASS $|\mathcal{N}_i|$ is a triple $(a, b, c) \in \mathbb{N}^3$ where
  - $a =$ number of non-rigid co-ordinates,
  - $b =$ number of arcs and
  - $c =$ number of unconstrained entry and exit co-ordinates.

Will it ever stop? — Size of a CVASS chain

- The size of a CVASS $|\mathcal{N}_i|$ is a triple $(a, b, c) \in \mathbb{N}^3$ where
  - $a =$ number of non-rigid co-ordinates,
  - $b =$ number of arcs and
  - $c =$ number of unconstrained entry and exit co-ordinates.
- The size of a CVASS chain C is $|C| = (|\mathcal{N}_1|, \ldots, |\mathcal{N}_w|) \in (\mathbb{N}^3)^*$.
- If we start with a CVASS chain of size $(a_1, b_1, c_1), (a_2, b_2, c_2), \ldots, (a_w, b_w, c_w)$ and expand it using one of the pro-
  cedures we saw earlier, the new CVASS chain will have size $(a_1, b_1, c_1), (a_{21}, b_{21}, c_{21}), \ldots, (a_{2r}, b_{2r}, c_{2r}), \ldots, (a_w, b_w, c_w)$.

Will it ever stop? — Size of a CVASS chain

- The size of a CVASS $|\mathcal{N}_i|$ is a triple $(a, b, c) \in \mathbb{N}^3$ where
  - $a =$ number of non-rigid co-ordinates,
  - $b =$ number of arcs and
  - $c =$ number of unconstrained entry and exit co-ordinates.
- The size of a CVASS chain C is $|C| = (|\mathcal{N}_1|, \ldots, |\mathcal{N}_w|) \in (\mathbb{N}^3)^*$.
- If we start with a CVASS chain of size $(a_1, b_1, c_1), (a_2, b_2, c_2), \ldots, (a_w, b_w, c_w)$ and expand it using one of the pro-
  cedures we saw earlier, the new CVASS chain will have size $(a_1, b_1, c_1), (a_{21}, b_{21}, c_{21}), \ldots, (a_{2r}, b_{2r}, c_{2r}), \ldots, (a_w, b_w, c_w)$.
- For any k between 1 and r, $(a_{2k}, b_{2k}, c_{2k}) <_{lex} (a_2, b_2, c_2)$.

The computation tree

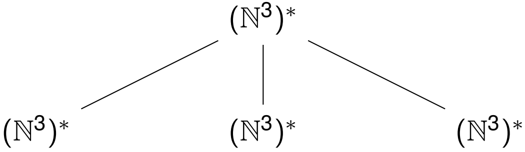


Figure: Computation tree

The computation tree

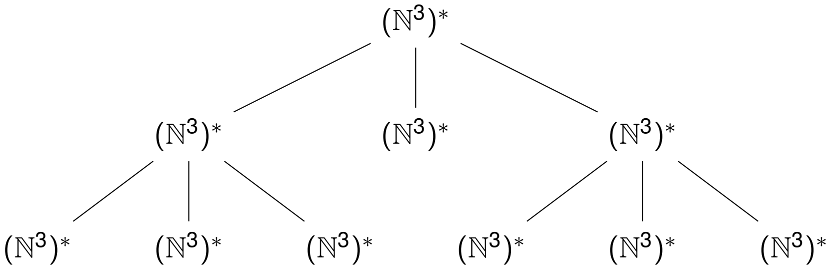


Figure: Computation tree

The computation tree

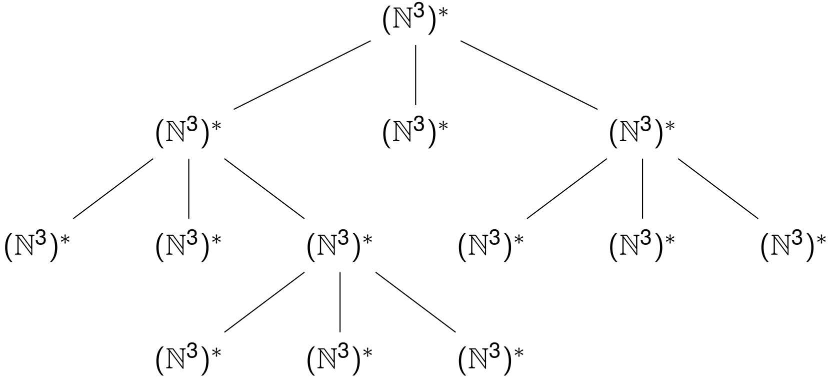


Figure: Computation tree

The computation tree

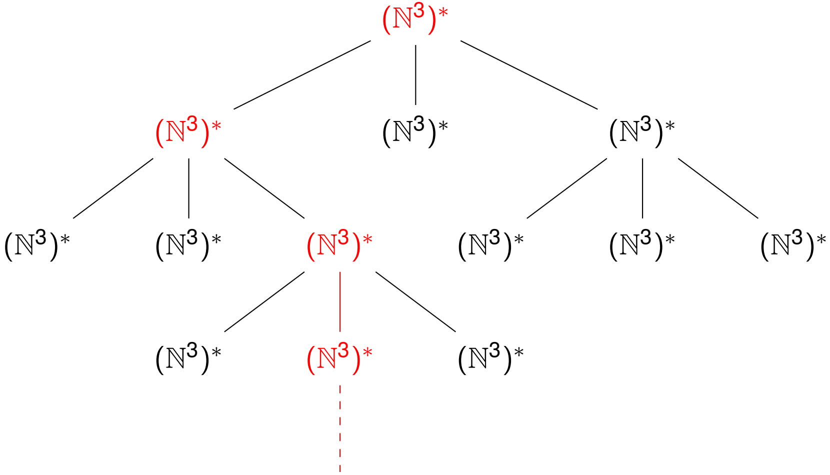


Figure: Computation tree

Computation tree - Contd. . .

(a, b, c)

Figure: Growth of the infinite path

Computation tree - Contd. . .

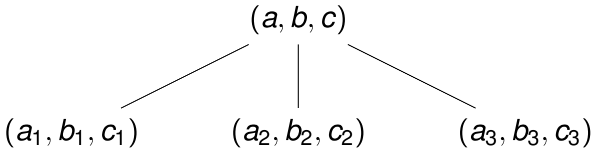


Figure: Growth of the infinite path

Computation tree - Contd...

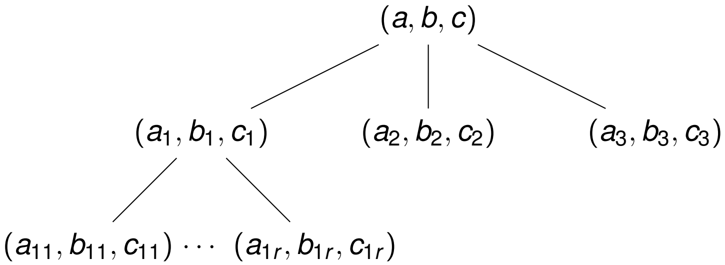


Figure: Growth of the infinite path

Computation tree - Contd. . .

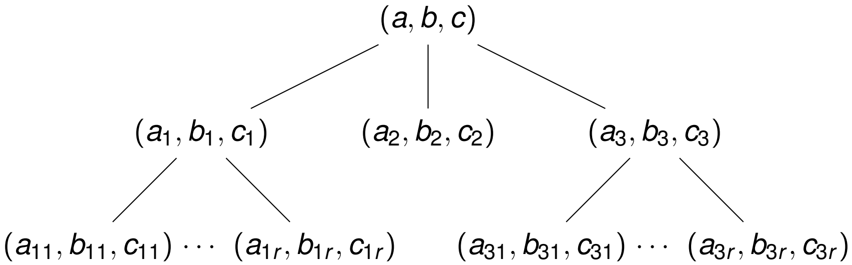


Figure: Growth of the infinite path

Computation tree - Contd. . .

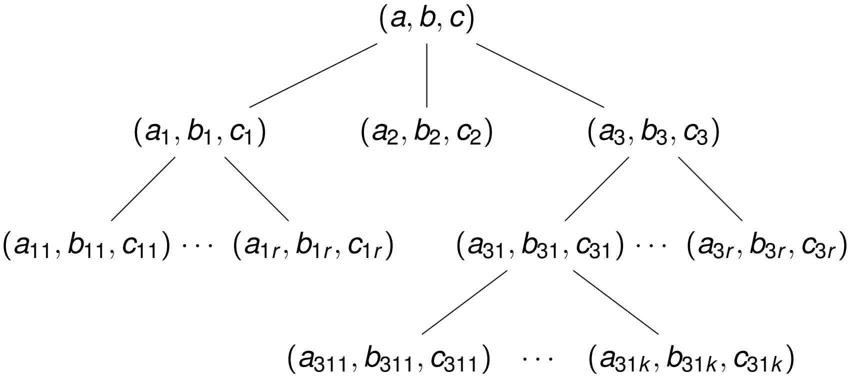


Figure: Growth of the infinite path

Computation tree - Contd. . .

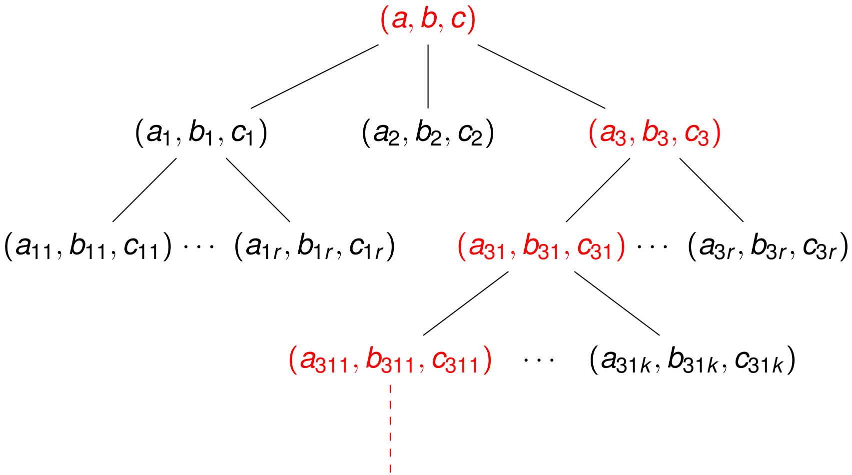


Figure: Growth of the infinite path

What if everything is infinite?

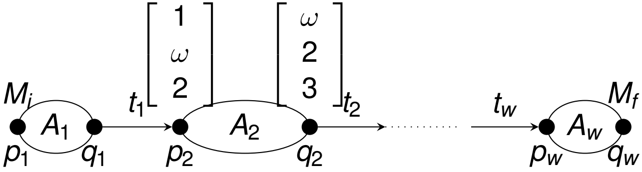


Figure: Everything infinite

- Kosaraju's condition θ: suppose there is a path from (p_1, M_i) to (q_1, M_f) and that
  - Every internal transition can be used unboundedly many times,
  - Every co-ordinate constrained at entry state is unbounded and
  - Every co-ordinate constrained at exit state is "reverse unbounded".

What if everything is infinite?

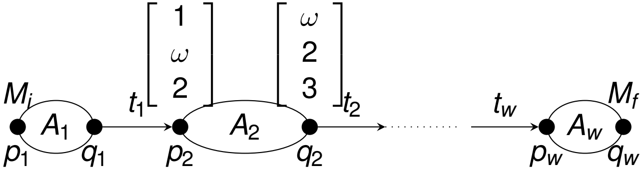


Figure: Everything infinite

- Kosaraju's condition θ: suppose there is a path from (p_1, M_i) to (q_1, M_f) and that
  - Every internal transition can be used unboundedly many times,
  - Every co-ordinate constrained at entry state is unbounded and
  - Every co-ordinate constrained at exit state is "reverse unbounded".
- No more finite things to hold on to. What do we do?

What if everything is infinite? The answer

- If everything is infinite, answer to the reachability question is yes!
- There is a path from (p_1, M_i) to (q_w, M_f), but co-ordinates may become negative while firing internal transitions.
- Since unconstrained co-ordinates can exceed any value, choose a path from (p_1, M_i) to (q_w, M_f) that assigns high enough values to all unconstrained co-ordinates.

What if everything is infinite? The answer

- If everything is infinite, answer to the reachability question is yes!
- There is a path from (p_1, M_i) to (q_w, M_f), but co-ordinates may become negative while firing internal transitions.
- Since unconstrained co-ordinates can exceed any value, choose a path from (p_1, M_i) to (q_w, M_f) that assigns high enough values to all unconstrained co-ordinates.
- What about constrained co-ordinates?

What if everything is infinite? The answer

- If everything is infinite, answer to the reachability question is yes!
- There is a path from (p_1, M_i) to (q_w, M_f), but co-ordinates may become negative while firing internal transitions.
- Since unconstrained co-ordinates can exceed any value, choose a path from (p_1, M_i) to (q_w, M_f) that assigns high enough values to all unconstrained co-ordinates.
- What about constrained co-ordinates?
- Pump them up! Use the self covering sequence to reach high enough values.
- Self covering sequence is not part of the path from (p_1, M_i) to (q_w, M_f), so it will cause some damage. Can we repair it?

What if everything is infinite? The answer

- If everything is infinite, answer to the reachability question is yes!
- There is a path from (p_1, M_i) to (q_w, M_f), but co-ordinates may become negative while firing internal transitions.
- Since unconstrained co-ordinates can exceed any value, choose a path from (p_1, M_i) to (q_w, M_f) that assigns high enough values to all unconstrained co-ordinates.
- What about constrained co-ordinates?
- Pump them up! Use the self covering sequence to reach high enough values.
- Self covering sequence is not part of the path from (p_1, M_i) to (q_w, M_f), so it will cause some damage. Can we repair it?
- Yes, by using the self destroying sequence!. This will need the fact that all transitions can be used unboundedly many times.

Detailed proof of Sufficiency theorem

- $E_i =$ Set of constrained entry co-ordinates at $\mathcal{N}_i$,
- $S_i =$ Set of constrained exit co-ordinates at $\mathcal{N}_i$ and
- $R_i =$ Set of rigid co-ordinates at $\mathcal{N}_i$.

The following morphism gives a semilinear set of extended commutative images of constrained paths from (P_1, M_i) and (q_w, M_f).

$$(entry[1], exit[1], \ldots, entry[w], exit[w], Parikh[1], \ldots, Parikh[w])$$

- First $2m$ co-ordinates gives the entry and exit co-ordinates of $\mathcal{N}_1$.
- n co-ordinates associated with $Parikh[1]$ gives the Parikh image of the path in $\mathcal{N}_1$.
- If a co-ordinate $j \notin E_1$, there will be corresponding non-zero entry in a period. Similarly for S_1.

Detailed proof of sufficiency theorem - Contd...

- Since all internal transitions can be used unboundedly often, every internal transition will have a corresponding non-zero entry in a period.
- Let **c** be a "constant" vector in the above semilinear set and **q** be the sum of all the "witnessing" periods.
- For any $k \in \mathbb{N}$, $\mathbf{c} + k\mathbf{q}$ is a vector corresponding some constrained walk from (p_1, M_i) to (q_w, M_f).
- We can assign large values to k to get large values at unconstrained co-ordinates and to use internal transitions large number of times.
- Now we concentrate on building a constrained positive path in $\mathcal{N}_i$.
- Let $\overline{\sigma(j)}$ denote the Parikh vector of the path in $\mathcal{N}_i$ given by $\mathbf{c} + j\mathbf{q}$.
- Let x_i (y_i) be the entry (exit) co-ordinate given by the constant vector **c**.

Detailed proof of sufficiency theorem - Contd...

- Let u_i (w_i) be the entry (exit) constraints given by $\mathbf{q}$.
- $(p_i, x_i) \xrightarrow{\sigma(0)} (q_i, y_i)$ and $(p_i, x_i + u_i) \xrightarrow{\sigma(1)} (q_i, y_i + w_i)$.
- $(p_i, x_i + u_i) \xrightarrow{\sigma(0)} (q_i, y_i + u_i) \xrightarrow{\sigma} (q_i, y_i + w_i)$, where $\overline{\sigma(1)} = \overline{\sigma(0)} + \overline{\sigma}$.
- $(q_i, u_i) \xrightarrow{\sigma} (q_i, w_i)$. $\mathit{effect}(\sigma) = w_i - u_i$.
- Let σ_1 be the pumping up sequence that pumps up constrained co-ordinates: $(p_i, x_i \overset{\sigma_1}{\Rightarrow}_{E_i} x_i + \Gamma_i)$, $\Gamma_i \restriction_{E_i} \geq (1, \ldots, 1)$.
- Let σ_4 be the pumping down sequence: $(q_i, y_i + \Delta_i) \overset{\sigma_4}{\Rightarrow}_{S_i} (q_i, y_i)$, $\Delta_i \restriction_{S_i} \geq (1, \ldots, 1)$.
- Let $\delta \geq 1$ be an integer greater than the absolute value of all co-ordinates of $\Gamma_i, \Delta_i, \overline{\sigma_1} + \overline{\sigma_4}$.
- Consider the sequence σ_3 such that $\overline{\sigma_3} = \delta\overline{\sigma} - \overline{\sigma_1} - \overline{\sigma_4}$.

Detailed proof of sufficiency theorem - Contd. . .

- Consider the "magic sequence of ℓ repetitions" $ms(\ell) = \sigma_1^\ell \sigma(0) \sigma_3^\ell \sigma_4^\ell$.
- If $k = \delta\ell$, then

$$(p_i, x_i + ku_i) \overset{\sigma_1^\ell}{\Rightarrow} (p_i, x_i + ku_i + \ell\Gamma_i) \overset{\sigma_0}{\Rightarrow} (q_i, y_i + ku_i + \ell\Gamma_i) \overset{\sigma_3^\ell}{\Rightarrow}$$

$$(q_i, y_i + kw_i + \ell\Delta_i) \overset{\sigma_4^\ell}{\Rightarrow} (q_i, y_i + kw_i).$$

- All the walks above can be made positive by choosing high enough value for k.

Conclusion

- Reachability in Petri nets is decidable.
- If some aspect of the net is bounded, unfold the net. Continue checking for boundedness of aspects in the expanded net.
- Termination of this process is shown by carefully defining a size and showing that it is well founded.
- If all aspects of the net are unbounded, conclude that answer to the reachability question is positive.
- The fact that all aspects of the net are unbounded can be expressed in terms of linear algebraic relations.

Thank you.

Questions?