Automates temporisés

– introduction par un néophyte –

Partie II / II – Complémentaire, vacuité et universalité

Mercredi 27 novembre 2002 – ÉNS Lyon

Jérôme DURAND-LOSE

jerome.durand-lose@ens-lyon.fr

MC2

LIP - ÉNS Lyon

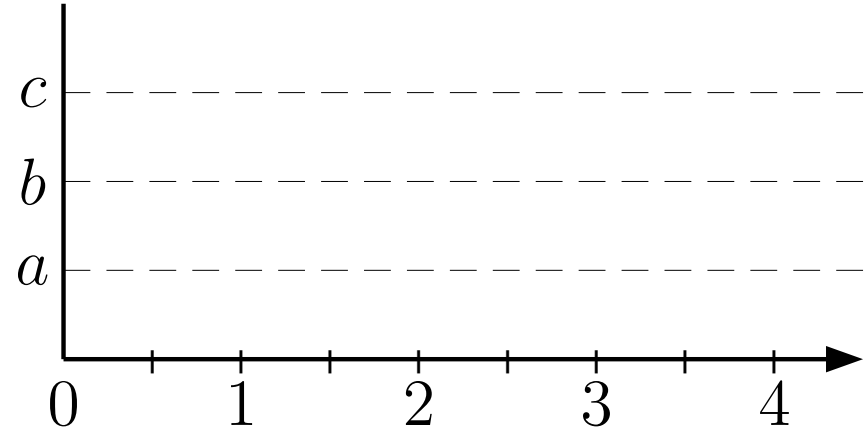
Plan

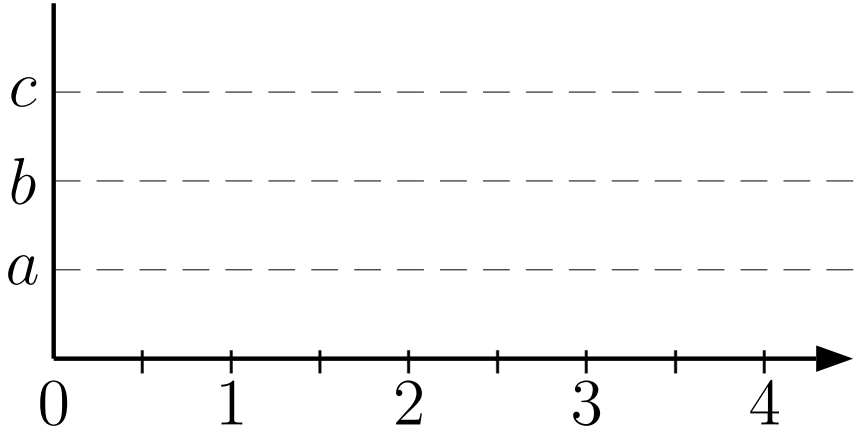
1. Automates (non temporisés)
   (a) Mots / langages / Automates finis
   (b) Extensions non temporisées (mots infinis)
2. Mots et langages temporisés
   (a) Définitions
   (b) Opérations, propriétés
3. Automates temporisés
   (a) Définitions, exemple
   (b) Automate des régions
4. Quelques complexités
   (a) Non clos par complémentaire
   (b) Vacuité est PSPACE-complet
   (c) Universalité est co-$\mathcal{R}.\mathcal{E}.$-complet

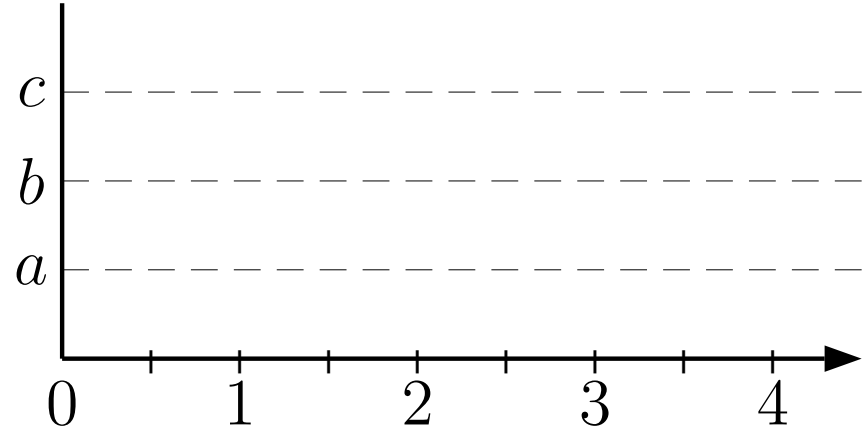
Mots temporisés

Mot non temporisé :

$abac$





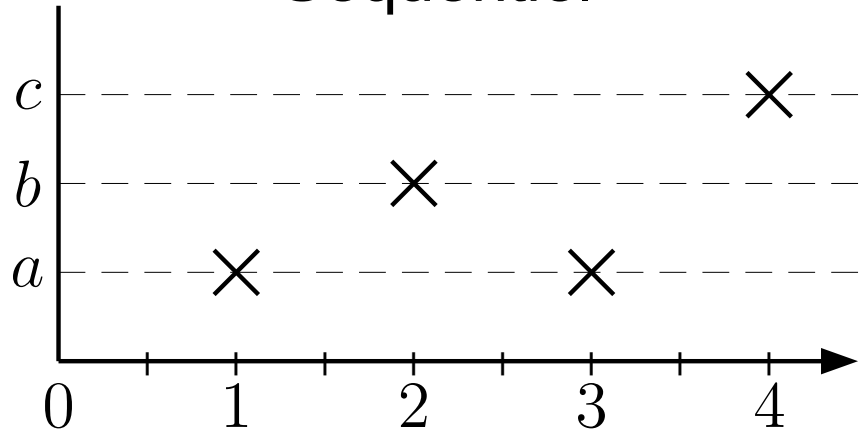


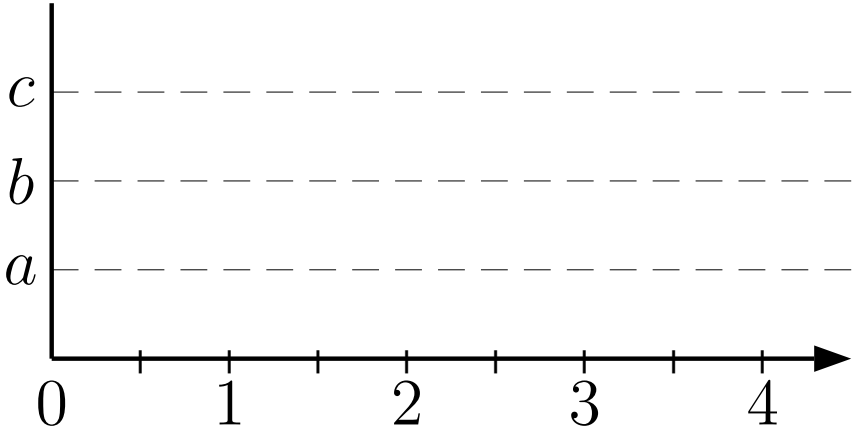
Mots temporisés

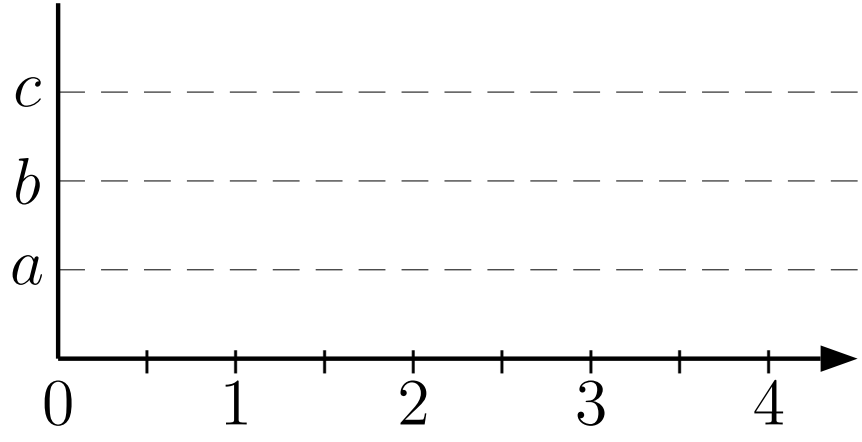
Mot non temporisé :

$abac$

Séquentiel





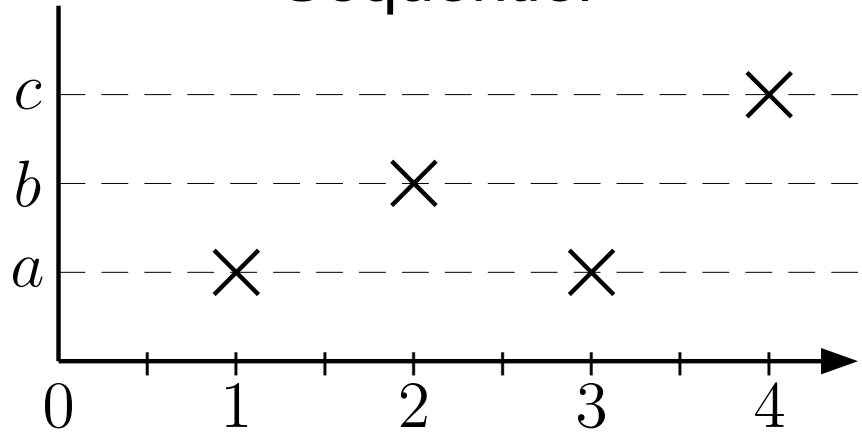


Mots temporisés

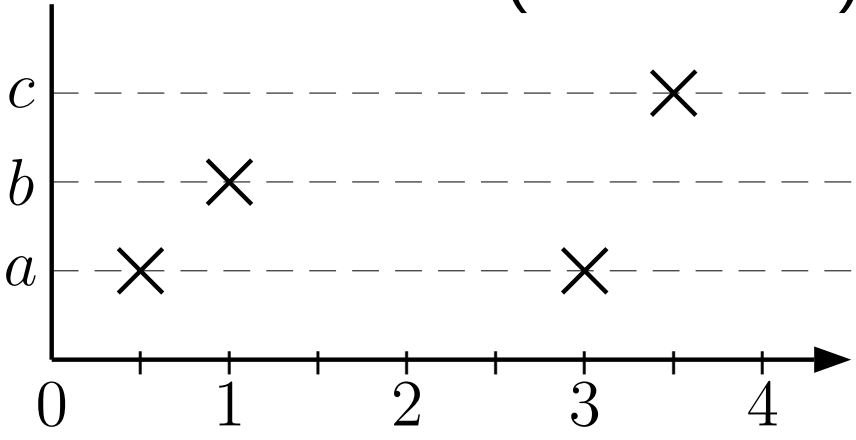
Mot non temporisé :

$abac$

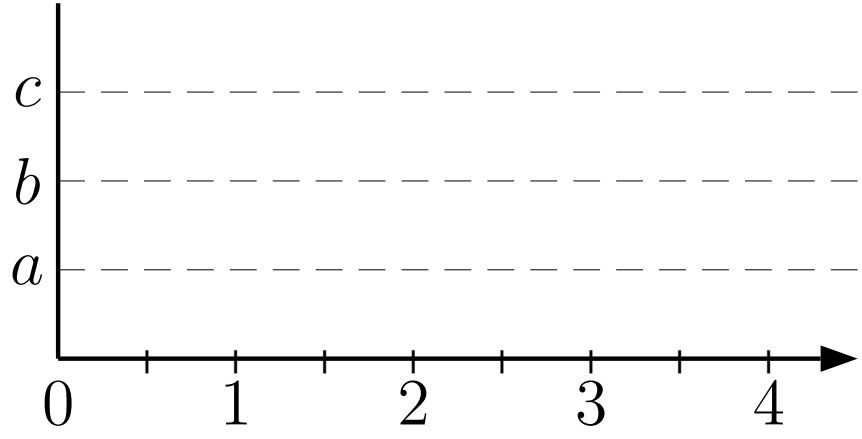
Séquentiel



Date-évènement (*time-event*)



$(a, 0.5)(b, 1)(a, 3)(c, 3.5)$



Langages temporisés et opérations

- *Durée* d'un mot
  $|(a, 0.5)(b, 1)(a, 3)(c, 3.5)| = 3.5$
- *Concaténation classique*
  $(a, 0.5)(b, 1)(a, 3)(c, 3.5) \bullet (a, 0.5)(b, 1)(a, 3)(c, 3.5)$
  $= (a, 0.5)(b, 1)(a, 3)(c, 3.5)(a, 4)(b, 4.5)(a, 6.5)(c, 7)$
- *Concaténation superposition* si consécutive / disjointe
  $(a, 0.5)(b, 1)(a, 3)(c, 3.5) \circ (a, 0.5)(b, 1)(a, 3)(c, 3.5)$ indéfini
  $(a, 0.5)(b, 1)(a, 3)(c, 3.5) \circ (a, 4)(b, 4.5)(a, 6.5)(c, 7)$
  $= (a, 0.5)(b, 1)(a, 3)(c, 3.5)(a, 4)(b, 4.5)(a, 6.5)(c, 7)$
- *Langage temporisé* : ensemble de mots temporisés
  Opérations
  - $\bullet$, $\circ$ et deux étoiles de KLEENE : $*$ et $\circledast$
  - Filtre sur durée : $\langle L \rangle_I = \{m \in l \mid |m| \in I\}$
    $\rightsquigarrow$ Aspects algébriques [Asarin et al., 2002]

Discret $\leftrightsquigarrow$ continu

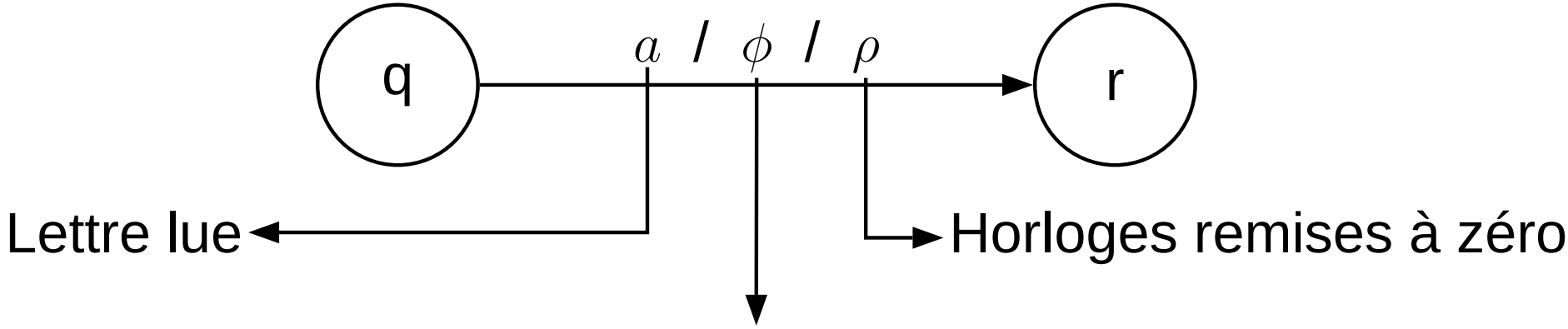
- Successions simultanées
  $(a, 0.5)(a, 1)(b, 1)(a, 1)(a, 2)\ldots$
  on peut l'exclure ou non
- Mots infinis pour comportement sans fin
  $\rightsquigarrow$ ω-mots temporisés
- Possibilité d'accumulation(s)
  $(a, 0.9)(a, .99)(b, .999)(a, .9999)(a, .99999)\ldots$
- Interdiction des *configurations Zénon*
  sinon $\rightsquigarrow$ mots temporisés sur des ordinaux
  [Bérard and Picaronny, 2000]

Automate temporisé

$\mathcal{A} = (\Sigma, Q, I, F, Z, \Delta)$

Z : ensemble fini d'*horloges*,

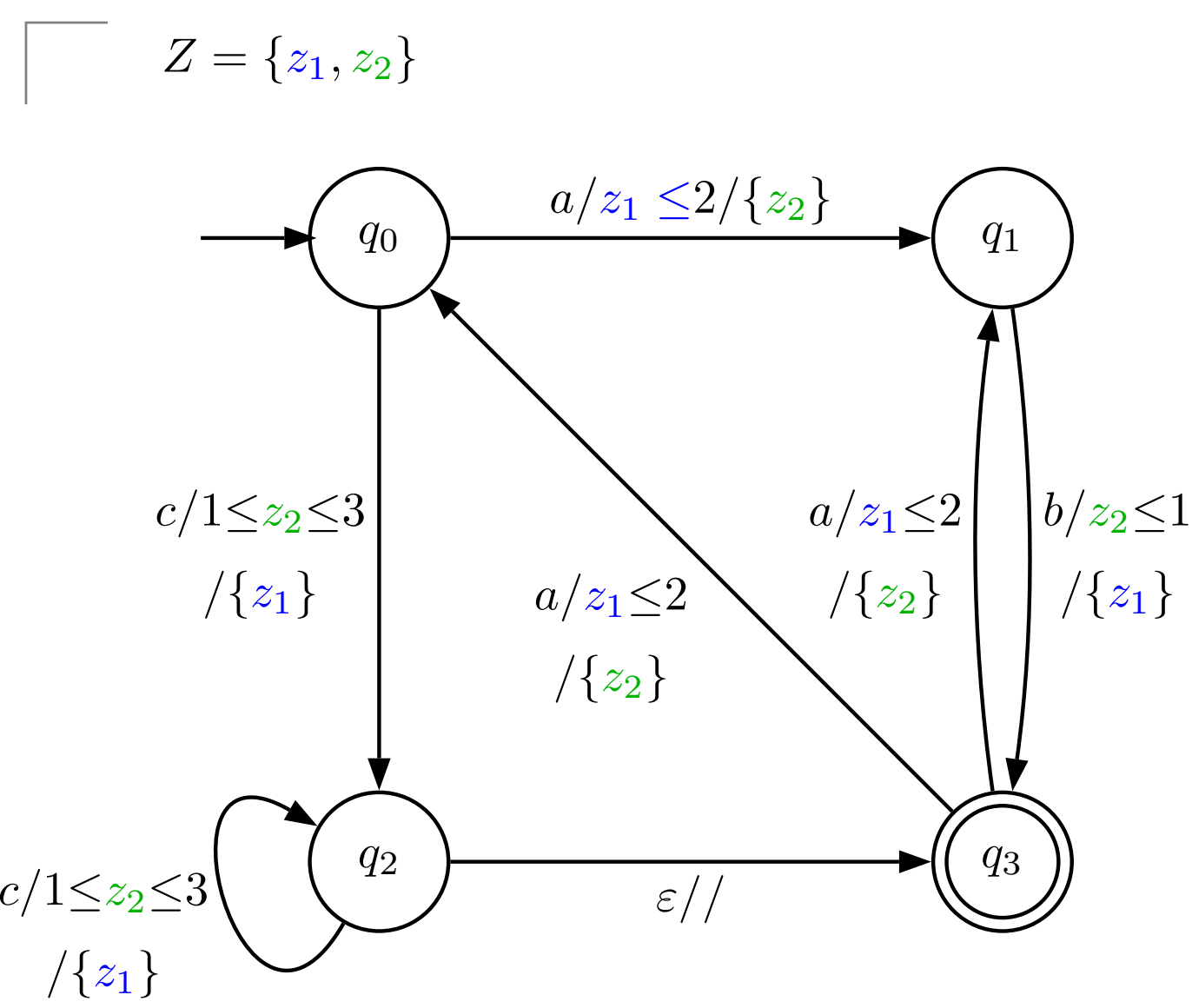
Δ : ensemble fini de transitions

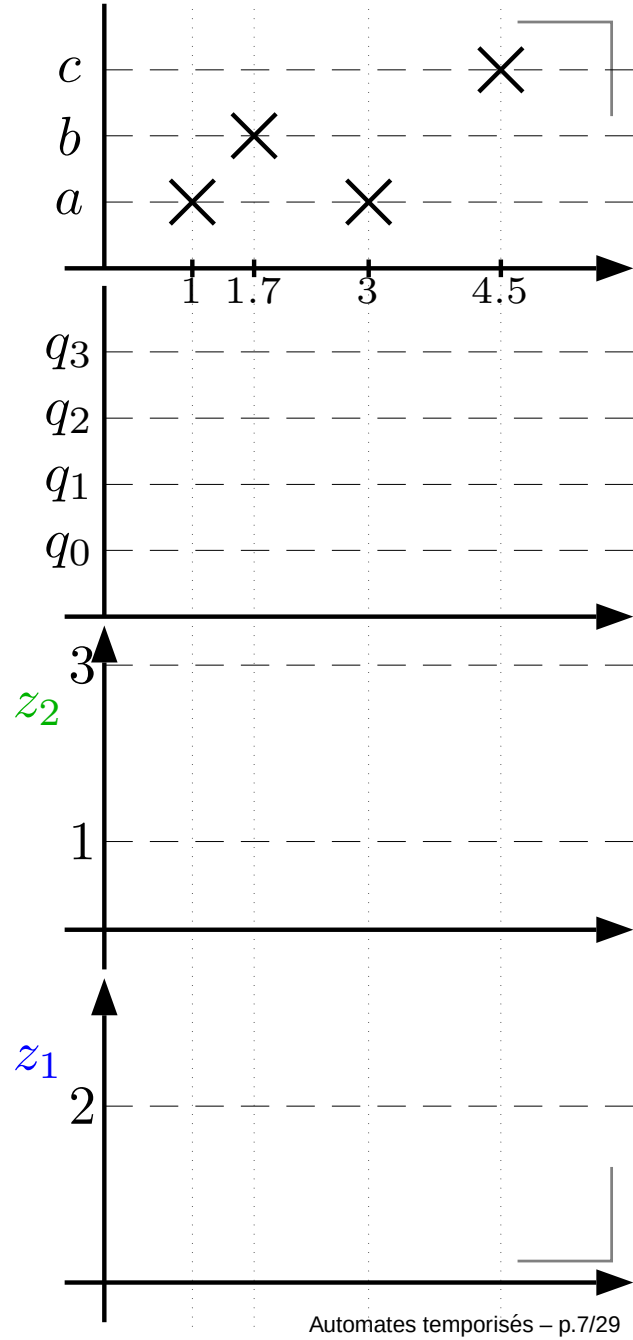


$z \leq c, \quad z < c, \quad \phi_1 \vee \phi_2, \quad \neg\phi$

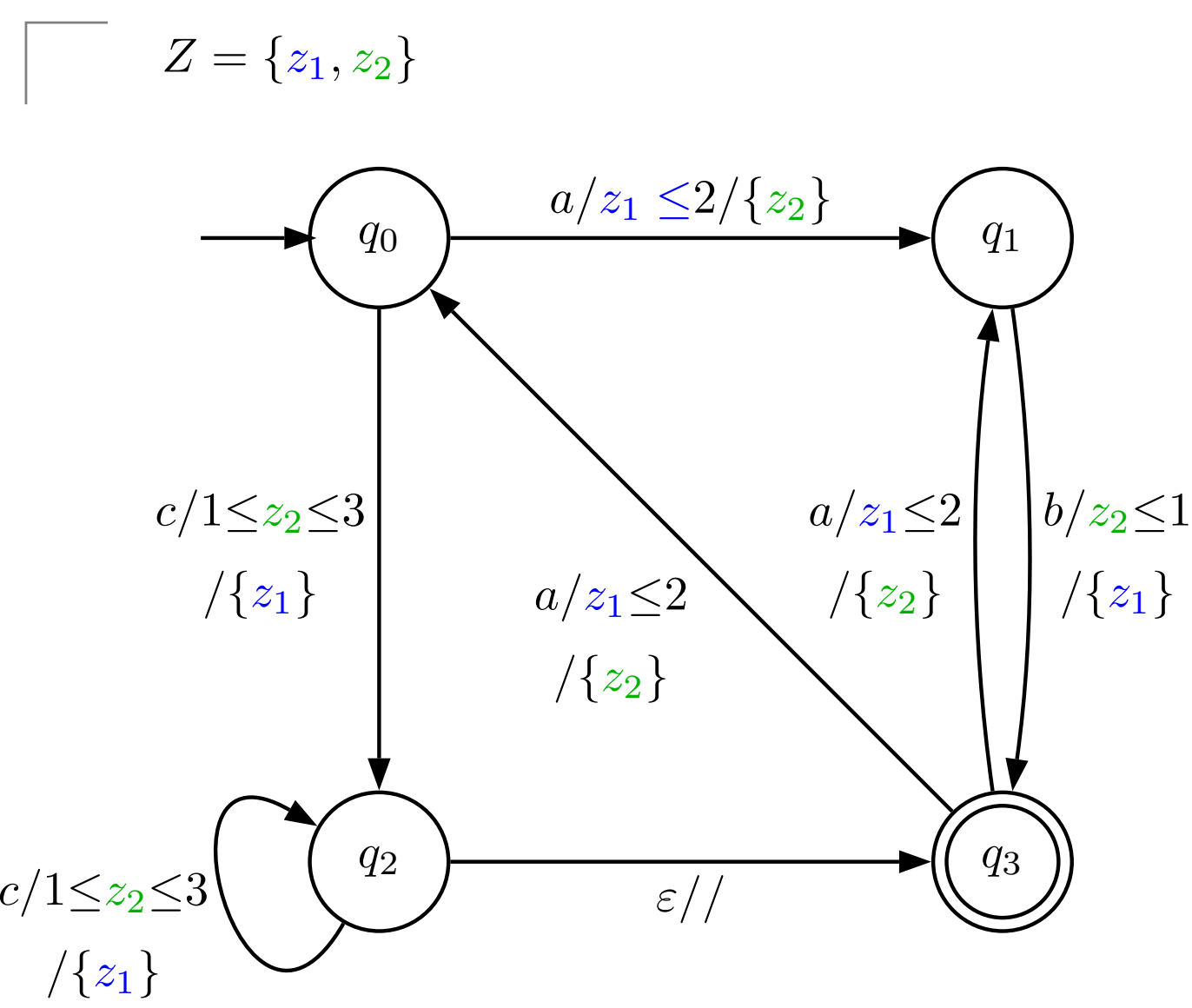
$c \in \mathbb{R}, \mathbb{Q}, \mathbb{N}$ (clauses plus complexes)

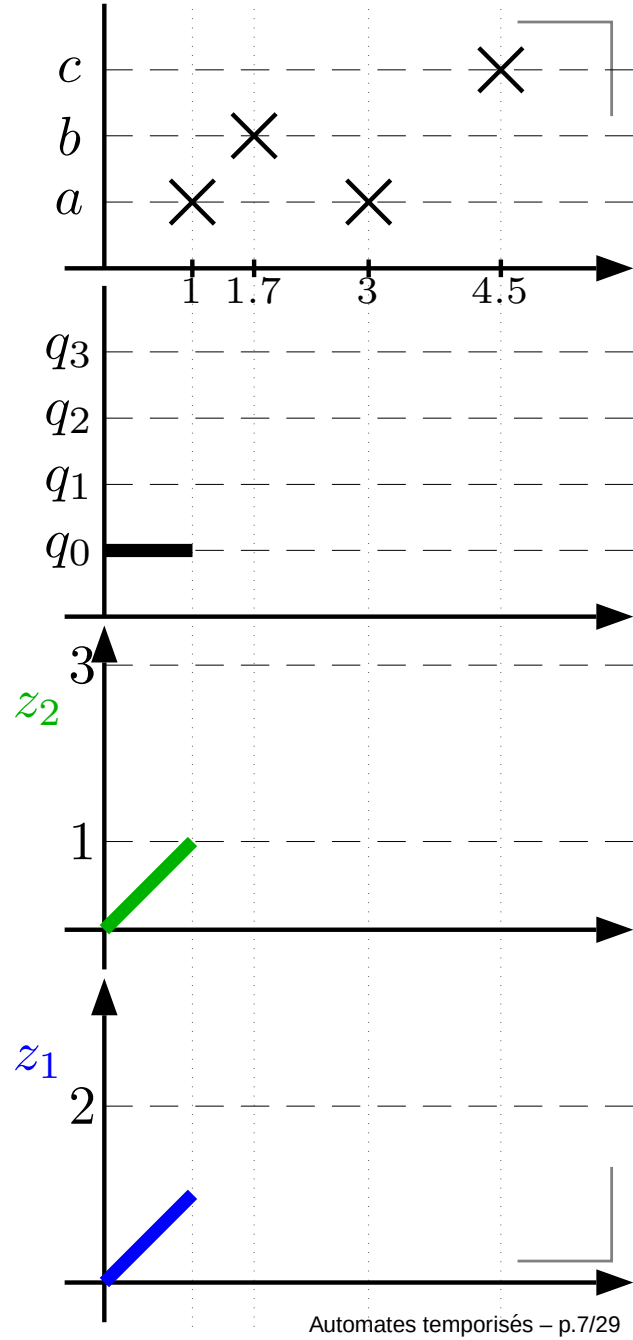
Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$

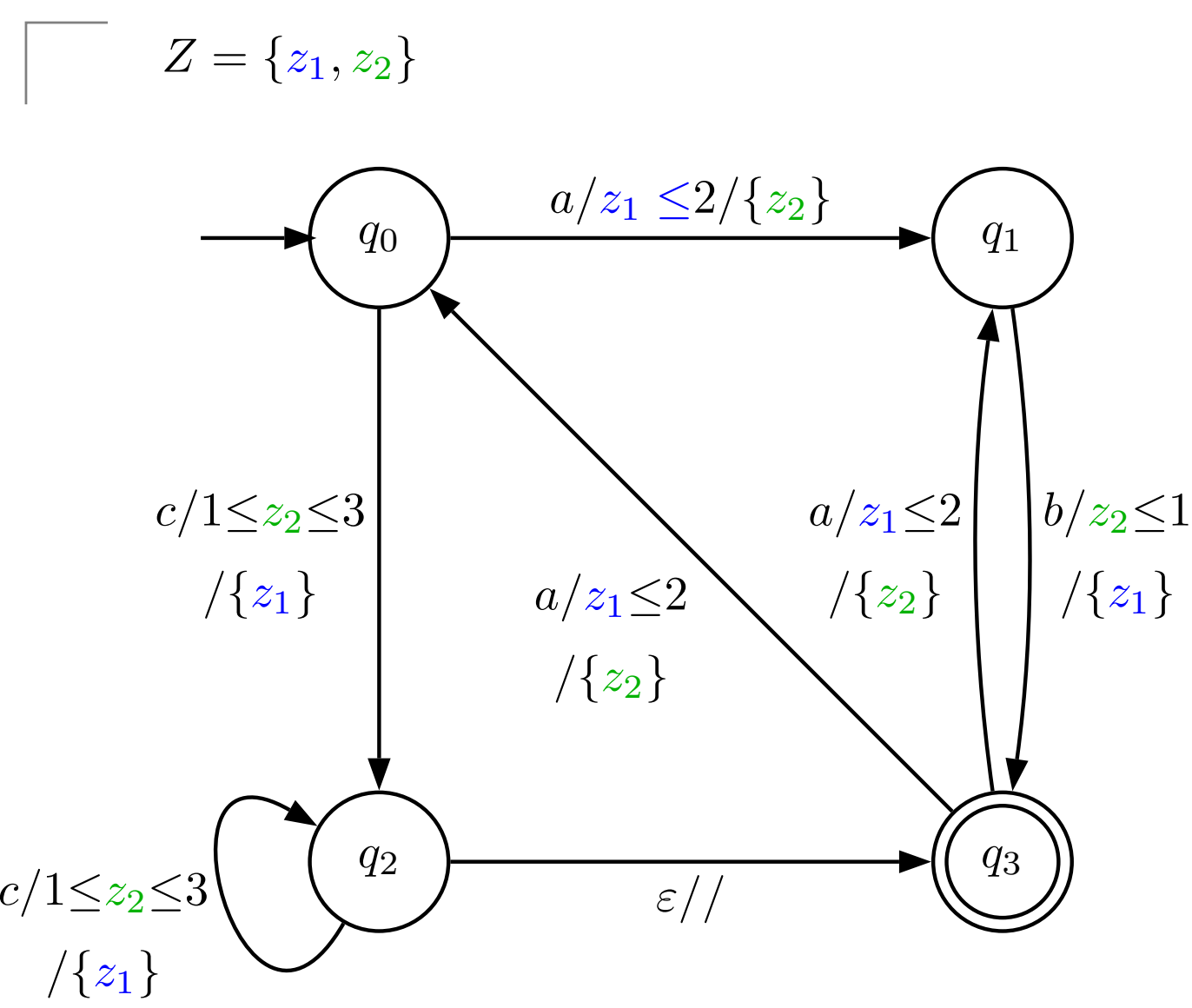

c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

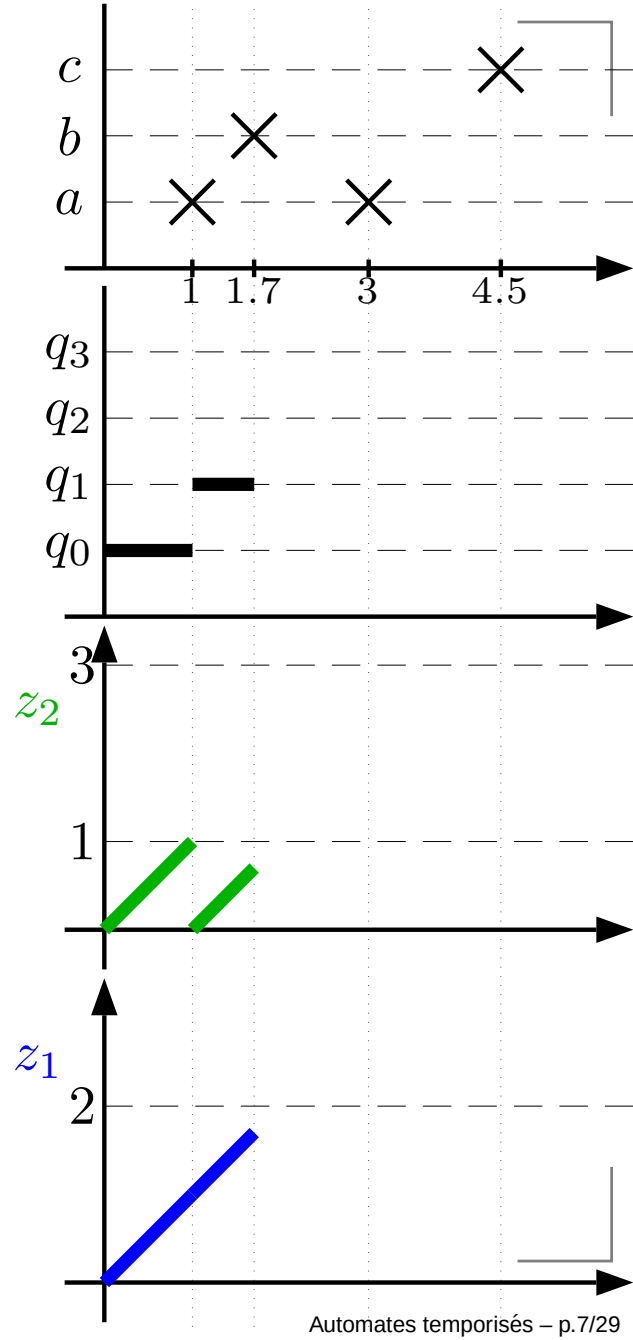
Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$

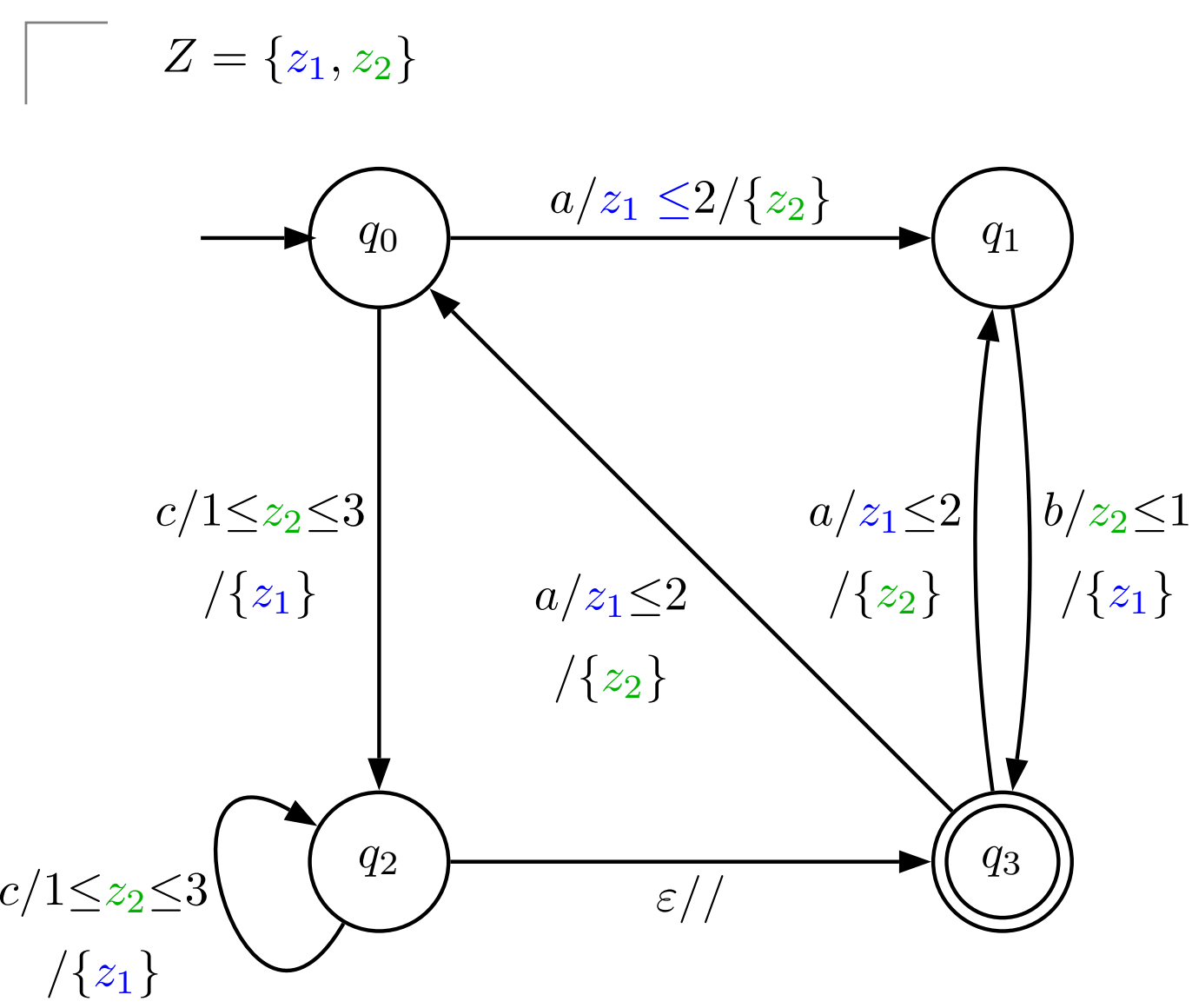

c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

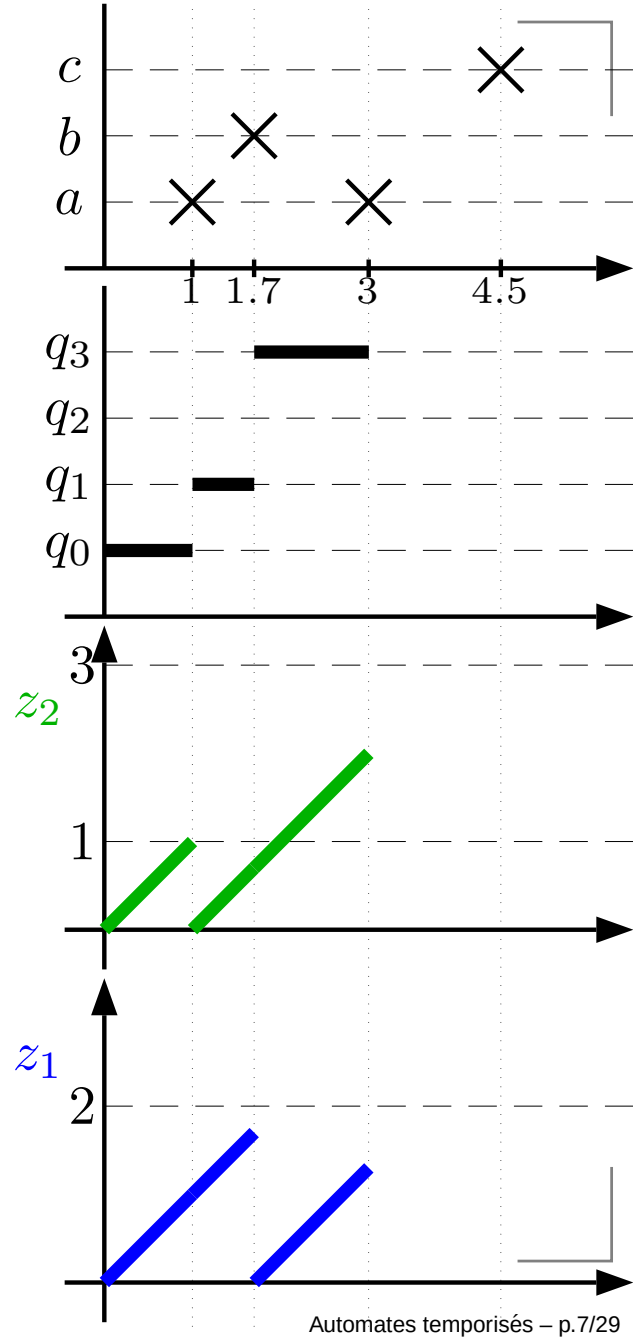
Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$

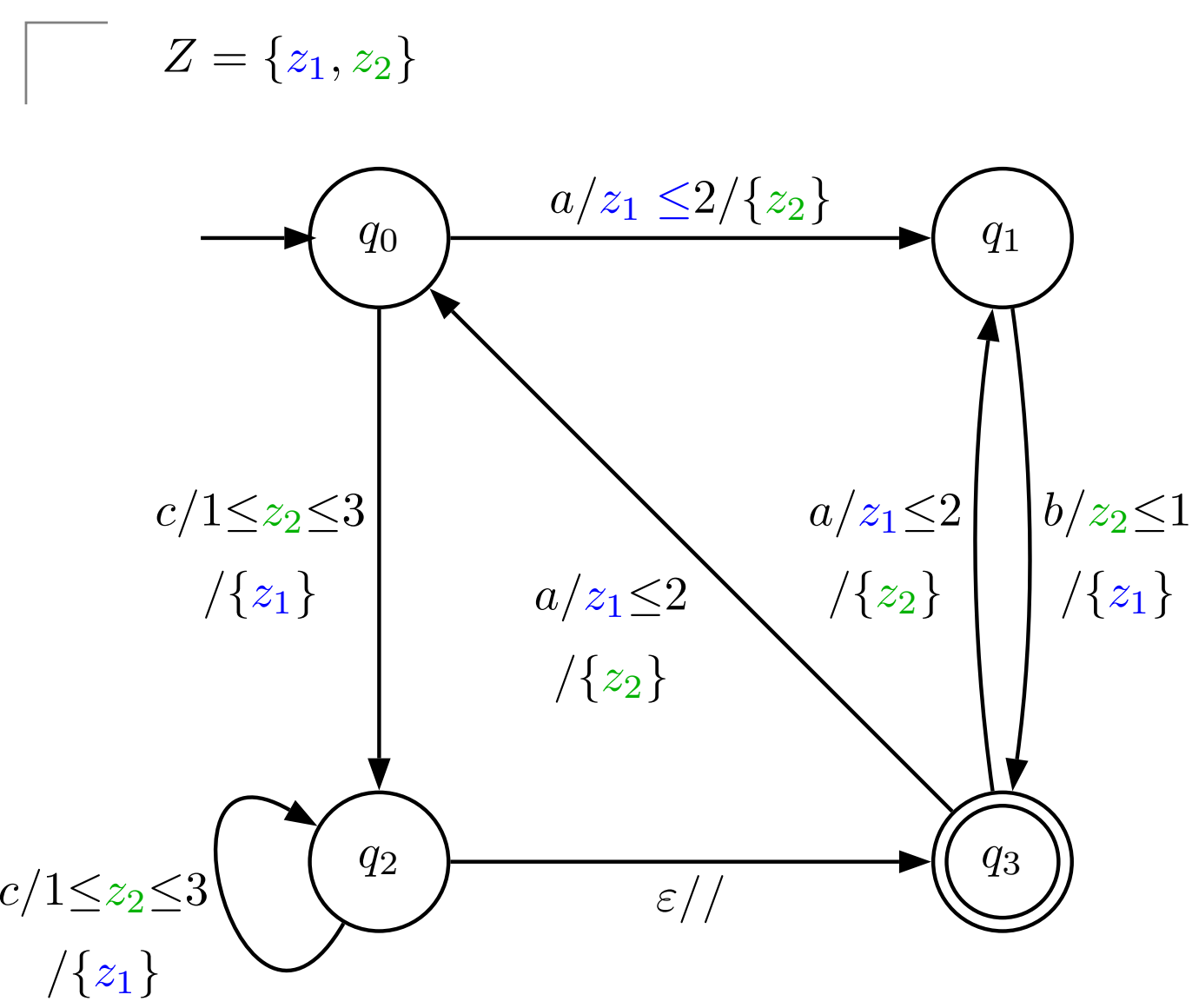

c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

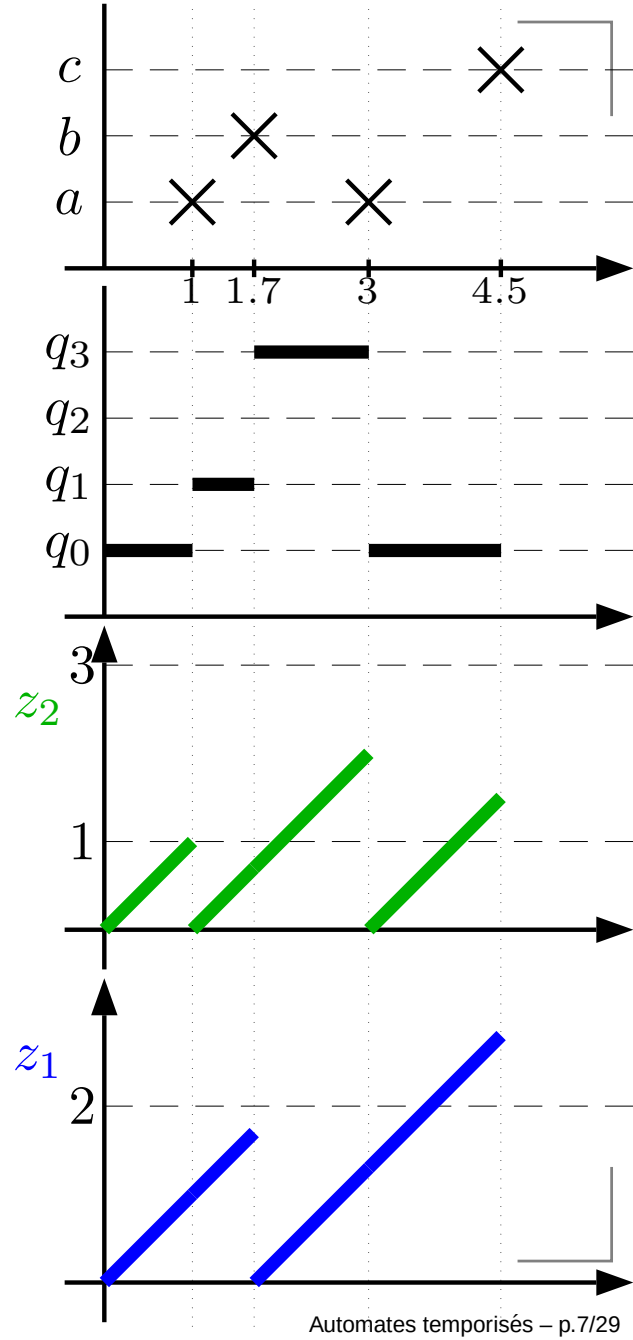
Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$

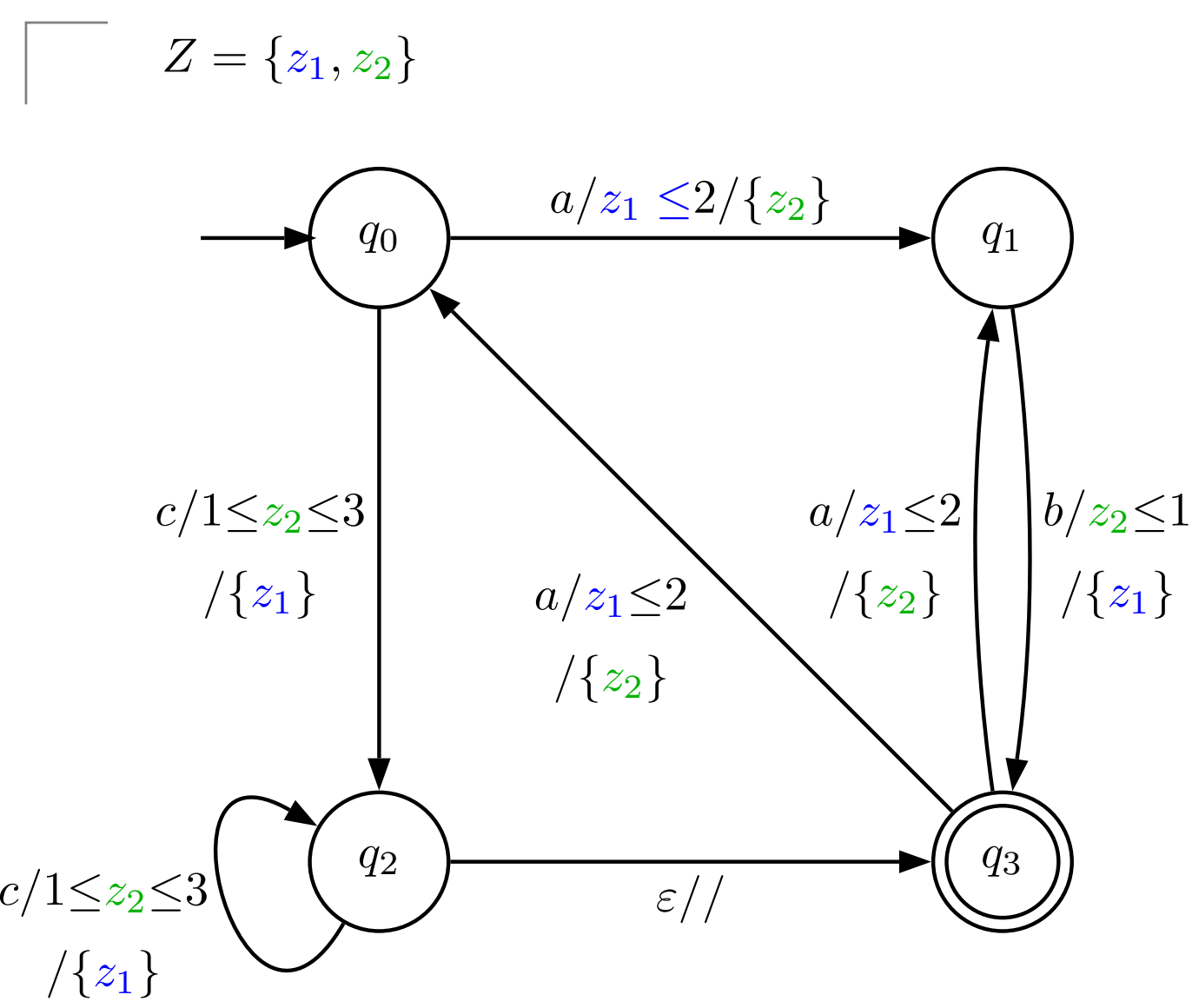

c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

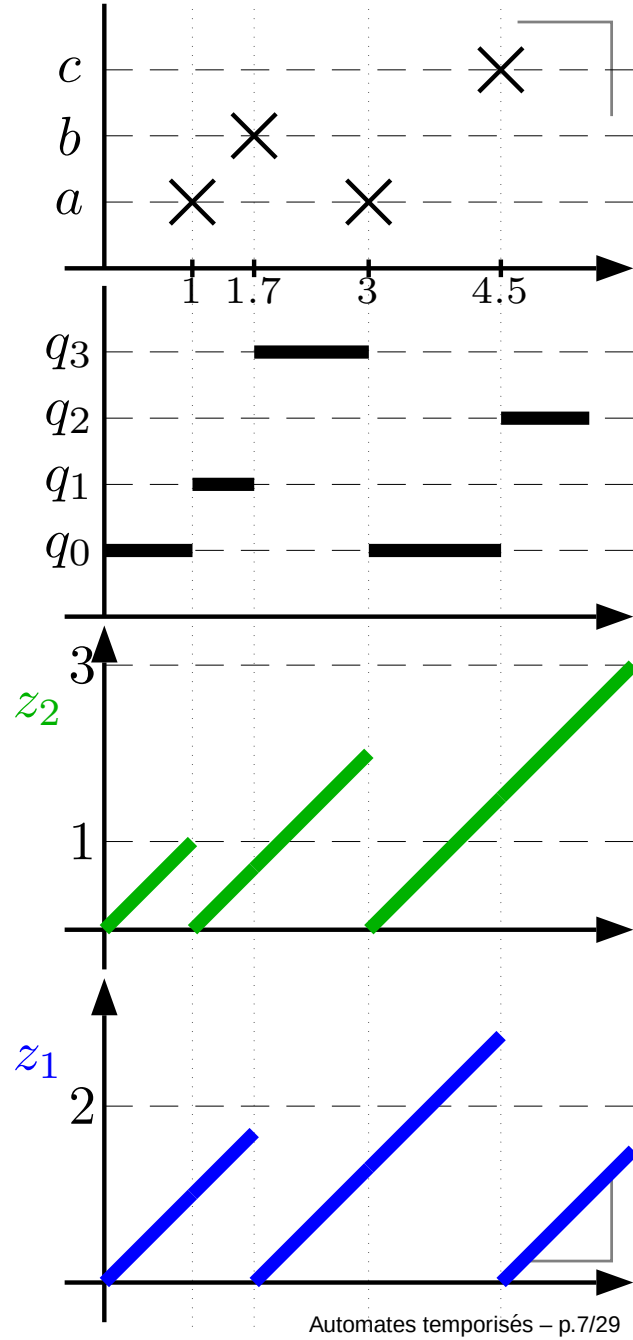
Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$

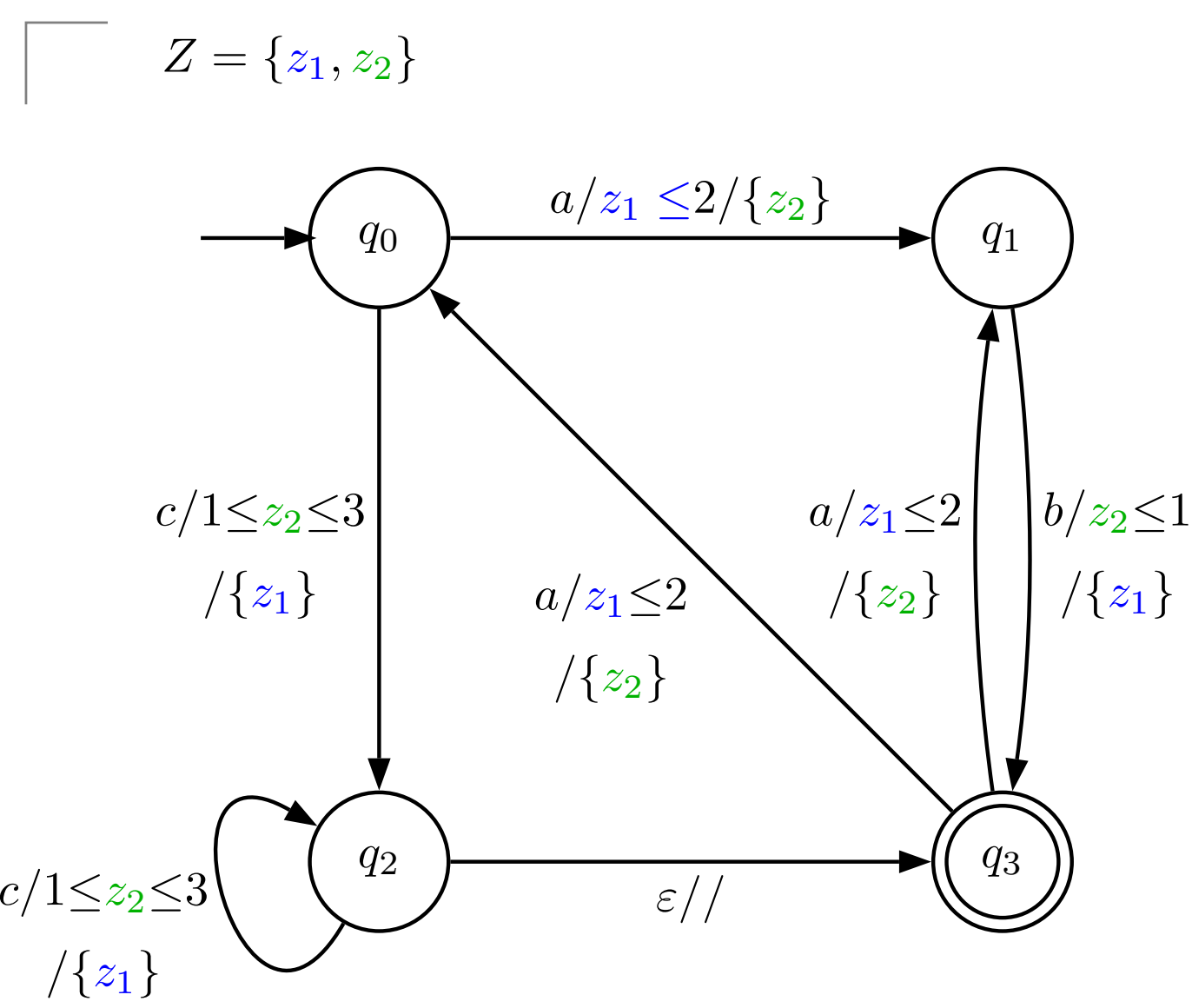

c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

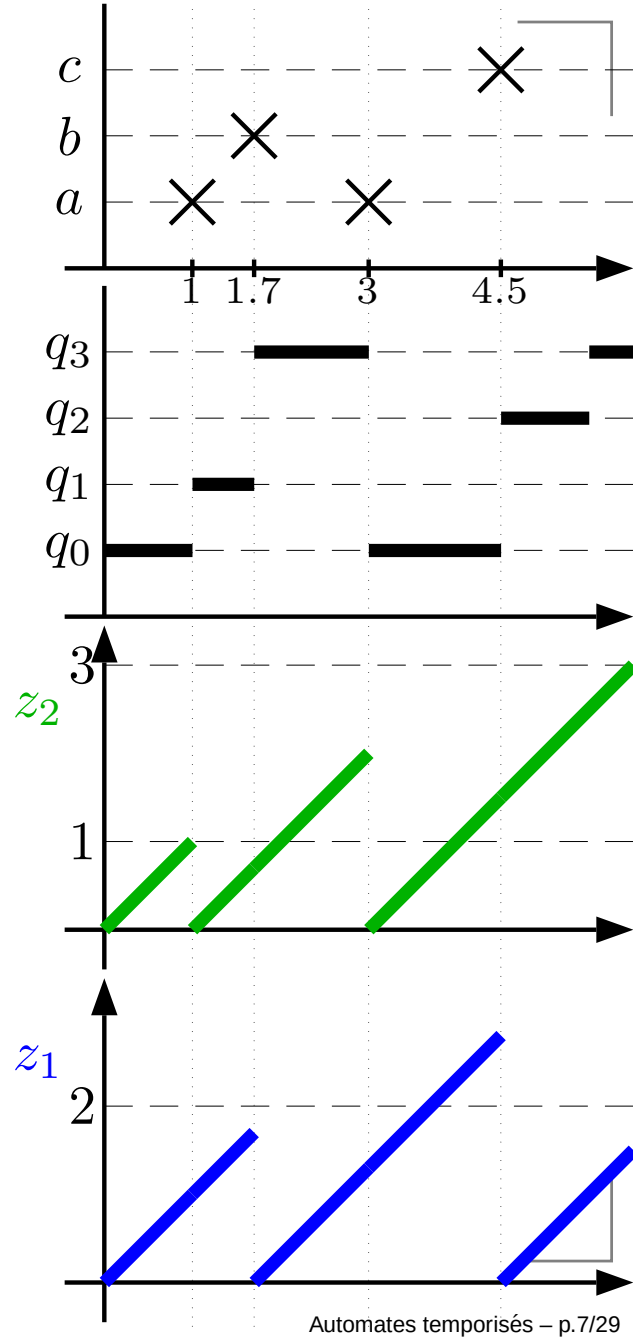
Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$


c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

Exemple


$Z = \{z_1, z_2\}$
q_0
q_1
q_2
q_3
$a/z_1 \leq 2/\{z_2\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$a/z_1 \leq 2$
$/\{z_2\}$
$b/z_2 \leq 1$
$/\{z_1\}$
$c/1 \leq z_2 \leq 3$
$/\{z_1\}$
$\varepsilon//$


c
b
a
1
1.7
3
4.5
q_3
q_2
q_1
q_0
z_2
3
1
z_1
2

Clos par...

- Union (facile car non déterministe)
- Intersection (produit habituel d'automate)
- Concaténation (recoller en remettant les horloges à 0)
- Superposition consécutive (recoller sans remise à 0)
- Itérations finies ($*$ et $\circledast$)
- Restriction sur la durée (une horloge en plus)

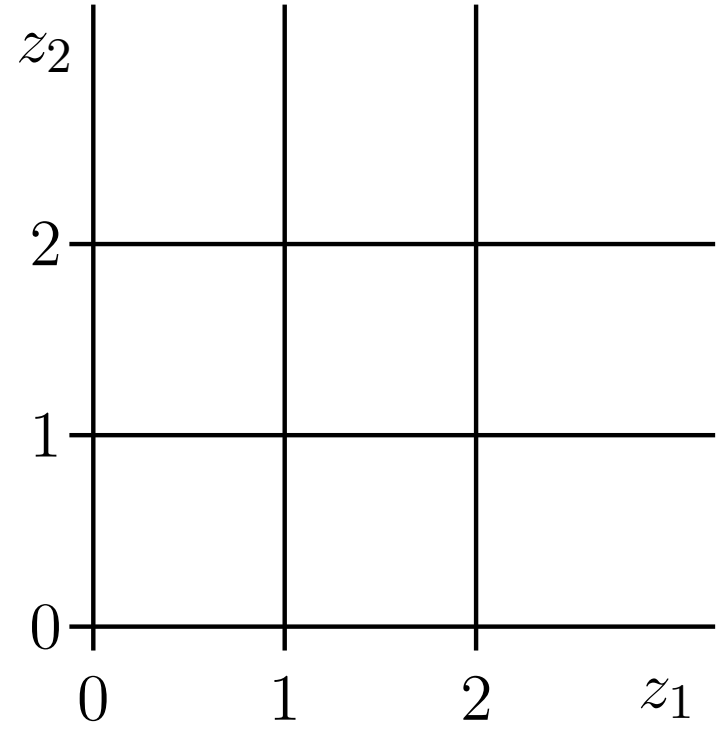
- Complément (?)

- Déterminisation (?)

- Automate minimal (?)

Se débarrasser du temps

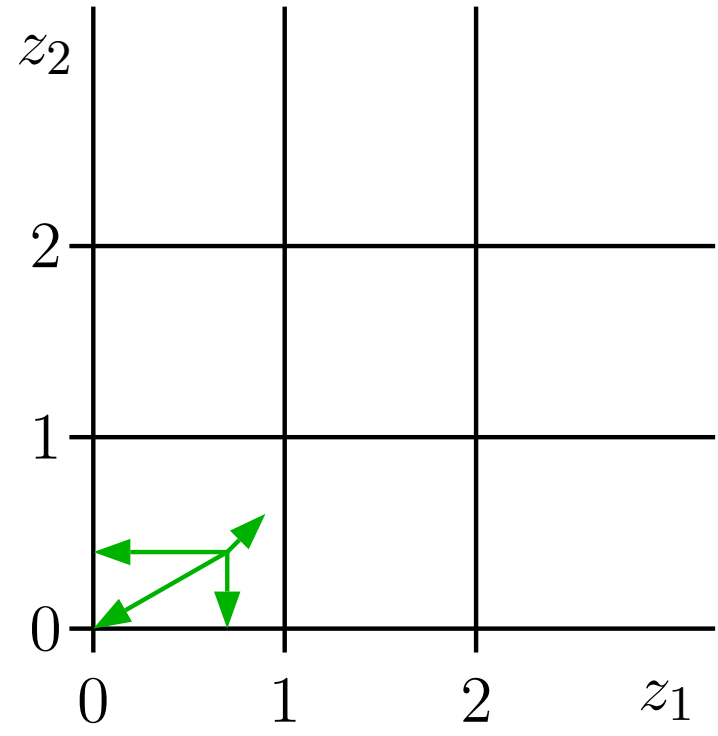
- On suppose les constantes des contraintes dans $\mathbb{Q}$
  En changeant d'échelle elles sont dans $\mathbb{N}$ (dans $[[1, C]]$)
- Découpage de l'espace des temps en *Clock regions*
  Lieux où les contraintes sont constantes
  - Être sur un entier ou entre 2
    $\{0\}, (0,1), \{1\}, (1,2), \{2\}, (2,\infty)$



- Augmentation exponentielle de la taille des données
  nombre de séparations de l'ordre de la plus grande constante... écrite en binaire

Se débarrasser du temps

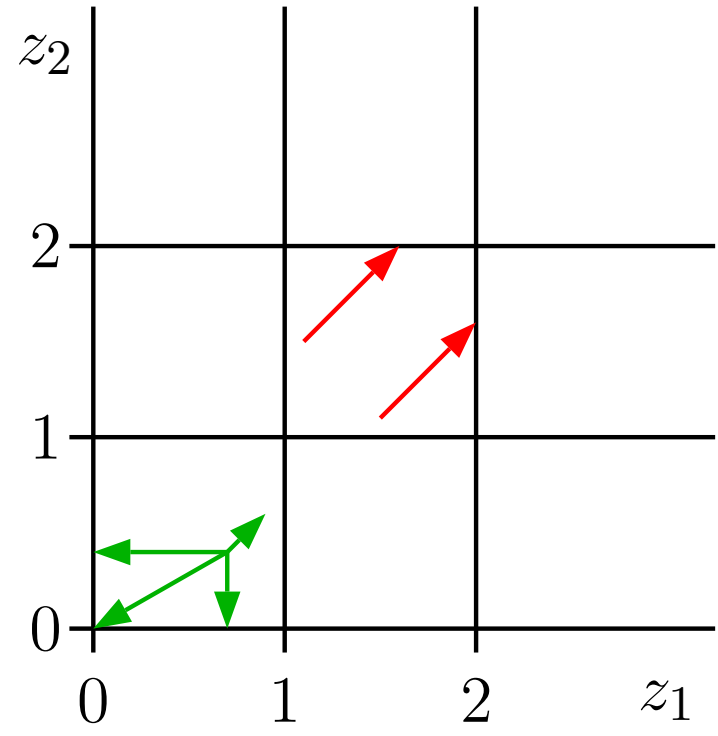
- On suppose les constantes des contraintes dans $\mathbb{Q}$
  En changeant d'échelle elles sont dans $\mathbb{N}$ (dans $[[1, C]]$)
- Découpage de l'espace des temps en *Clock regions*
  Lieux où les contraintes sont constantes
  - Être sur un entier ou entre 2
    $\{0\}$, $(0,1)$, $\{1\}$, $(1,2)$, $\{2\}$, $(2,\infty)$
  - Mouvements autorisés :
    - projection(s) (remise(s) à 0)
    - le temps avance suivant $\overline{1}$



- Augmentation exponentielle de la taille des données
  nombre de séparations de l'ordre de la plus grande constante... écrite en binaire

Se débarrasser du temps

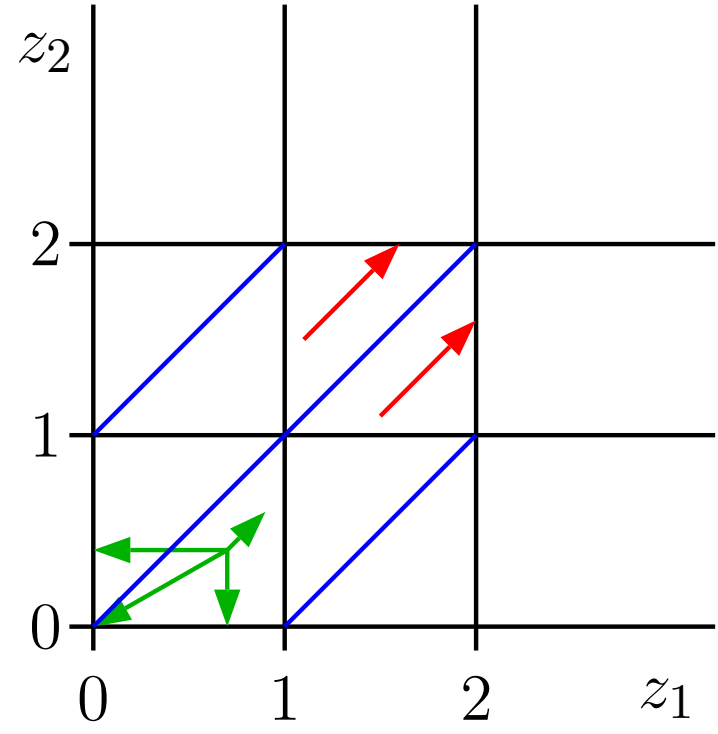
- On suppose les constantes des contraintes dans $\mathbb{Q}$
  En changeant d'échelle elles sont dans $\mathbb{N}$ (dans $[[1, C]]$)
- Découpage de l'espace des temps en *Clock regions*
  Lieux où les contraintes sont constantes
  - Être sur un entier ou entre 2
    $\{0\}, (0,1), \{1\}, (1,2), \{2\}, (2,\infty)$
  - Mouvements autorisés :
    - projection(s) (remise(s) à 0)
    - le temps avance suivant $\overline{1}$
  - Problème suivant $\overline{1}$!



- Augmentation exponentielle de la taille des données
  nombre de séparations de l'ordre de la plus grande constante... écrite en binaire

Se débarrasser du temps

- On suppose les constantes des contraintes dans $\mathbb{Q}$
  En changeant d'échelle elles sont dans $\mathbb{N}$ (dans $[[1, C]]$)
- Découpage de l'espace des temps en *Clock regions*
  Lieux où les contraintes sont constantes
  - Être sur un entier ou entre 2
    $\{0\}, (0,1), \{1\}, (1,2), \{2\}, (2,\infty)$
  - Mouvements autorisés :
    - projection(s) (remise(s) à 0)
    - le temps avance suivant $\overline{1}$
  - Problème suivant $\overline{1}$!
  - Pré-ordre total sur
    les parties fractionnaires
    $Frac(z_1) \leq = \geq Frac(z_2)$



- Augmentation exponentielle de la taille des données
  nombre de séparations de l'ordre de la plus grande constante... écrite en binaire

Region automaton

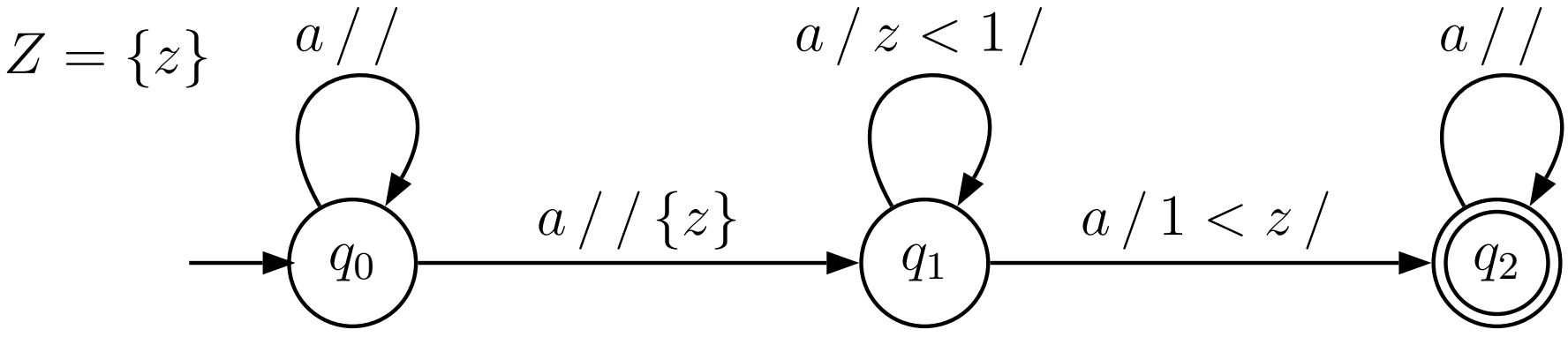
- Soit $\mathcal{R}$, l'ensemble des régions
- Les régions α et β se *succèdent*

$$\alpha \preccurlyeq \beta \text{ ssi } \alpha + \lambda\overline{1} \subseteq \beta \text{ avec } \lambda \text{ positif}$$

- $Q' = Q' \times \mathcal{R}$
  $I' = I = \times\{\overline{0}\}$
  $F' = F = \times\mathcal{R}$
- $(q, \alpha) \xrightarrow{a} (r, \beta)$ ssi $\begin{cases} q \xrightarrow{a/\phi/\rho} r \\ \exists \gamma \in \mathcal{R} \begin{cases} \phi \text{ vrai sur } \gamma \\ Reset(\gamma, \rho) = \beta \end{cases} \end{cases}$
- Reconnaît exactement le langage dé-temporisé qui est donc rationnel

Clôture par complémentation ?

- $\Sigma = \{a\}$
- Soit L_0 le langage accepté par l'automate suivant :



Il existe un a (au moins 1 avant la fin du mot) sans a 1 après

- Son complémentaire peut-il être reconnu par un automate temporisé ?
  Non, par l'absurde

Complémentaire de L_0

- $\complement L_0$: pour tout a, un a 1 après (si non en dehors du mot)
- Soit $\mathcal{A}$ un automate temporisé reconnaissant $\complement L_0$
- Soit
  $m_k = (a,0)(a,\frac{1}{k^2})(a,\frac{2}{k^2})\ldots(a,\frac{1}{k})(a,1)(a,1+\frac{1}{k^2})\ldots(a,1+\frac{1}{k})$
  $k >$ le nombre d'horloges et $\frac{1}{k} <$ la plus petite constante
- Il existe un chemin d'acceptation pour m_k
  soit i tel que $(a,\frac{i}{k^2})$ ne remet à 0 aucune horloge non remise à 0 sur $]\frac{i}{k^2},1[$
- On remplace $(a,\frac{i}{k^2})$ par $(a,\frac{i}{k^2}+\frac{1}{k^3})$,
  le mot est toujours reconnu par le même chemin
  mais il n'appartient pas à $\complement L_0$

Conséquences

- L'ensemble des langages reconnus par automates temporisés n'est pas clos par complément
- Par contre les langages reconnus par automates temporisés *déterministes* sont clos par complément
- Les automates temporisés déterministes reconnaissent strictement moins de langages

Langage reconnu vide ?

Instance (Vacuité)

$\mathcal{A}$: automate temporisé

Question

$\mathcal{L}(\mathcal{A})$ est-il vide ?

- Temps polynomial pour un non temporisé
  Donc au pire en temps exponentiel en passant par les régions
- On peut rester en espace polynomial en travaillant en *non-déterministe* sur l'expression *implicite* de l'automate par région
- NP-SPACE = D-PSPACE= PSPACE (Savitch 70)
  Au pire déterministe en espace polynomial

PSPACE-difficile

Instance (PSPACE-complet [Hopcroft and Ullman, 1979]

M : machine de Turing

m : entrée pour M

Question

M accepte-t-il m sans que la tête sorte de m ?

Réduction : le seul mot accepté code toutes les configurations du départ à l'acceptation

$$\Sigma = \Gamma \cup (\Gamma \times Q) \cup \{a_0\} = \{a_0, a_1, \ldots a_k\}$$

Codage des configurations

$$\begin{array}{|c|c|c|c|c|c|c|} \hline \gamma_1^j & \gamma_2^j & \gamma_3^j & \dots & \gamma_i^j & \dots & \gamma_n^j \\ \hline \end{array} = \sigma_1^j\ \sigma_2^j \dots\ \sigma_n^j \qquad (q^j)$$

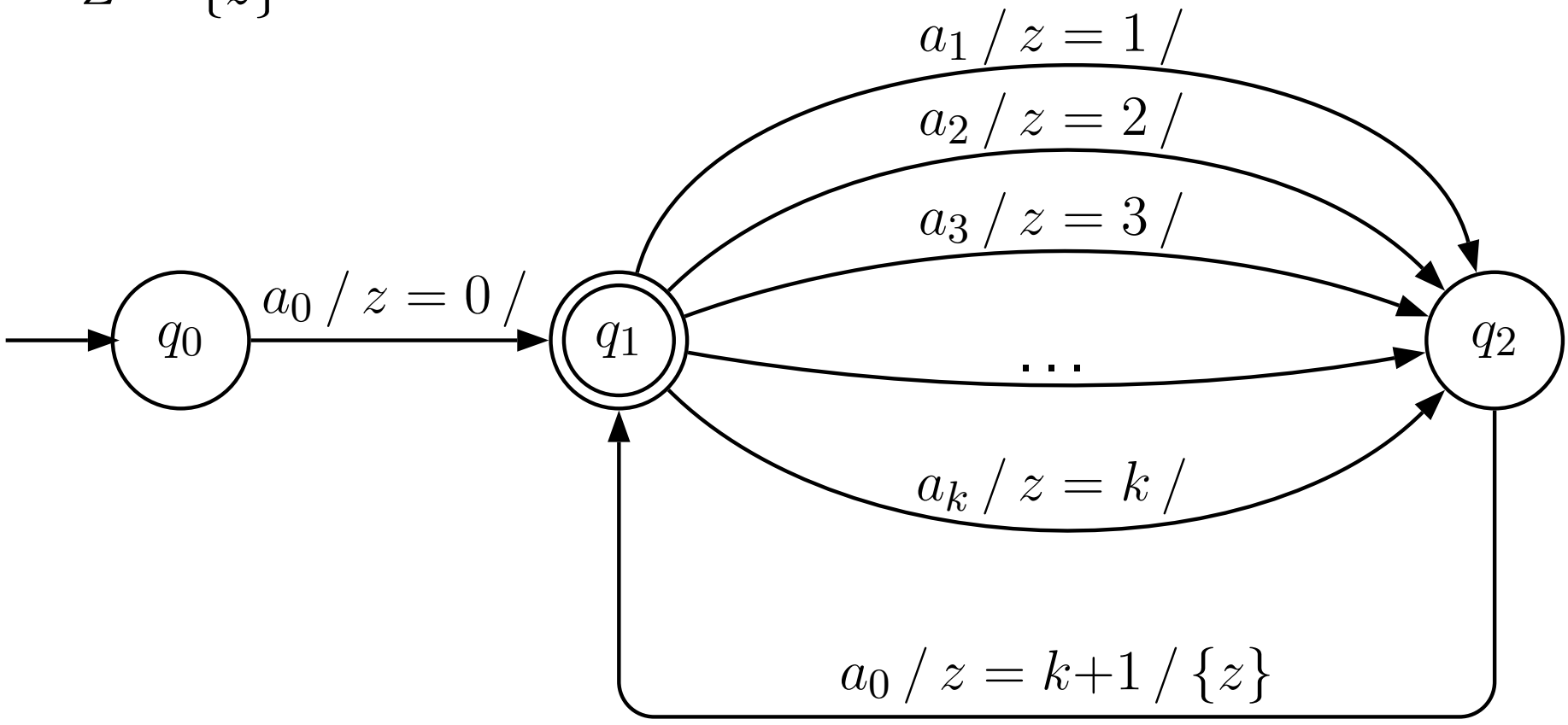
$$\overbrace{a_0\,\sigma_1^0\,a_0}^{k+1}\ \overbrace{\sigma_2^0\,a_0\,\sigma_3^0\,a_0}^{k+1}\ \dots\ \overbrace{a_0\,\sigma_n^0\,a_0}^{k+1}\ \overbrace{\sigma_1^1\,a_0\,\sigma_2^1\,a_0}^{k+1}\ \dots\ a_0\,\sigma_n^1\,a_0\,\sigma_1^2\,a_0\ \dots$$

Il suffit d'une seule horloge z pour vérifier cela!

$$\dots\ \overbrace{a_0\ a_l}^{l}\ \overbrace{a_0\ a_m}^{m}\ \dots$$

On peut le vérifier avec la même horloge z

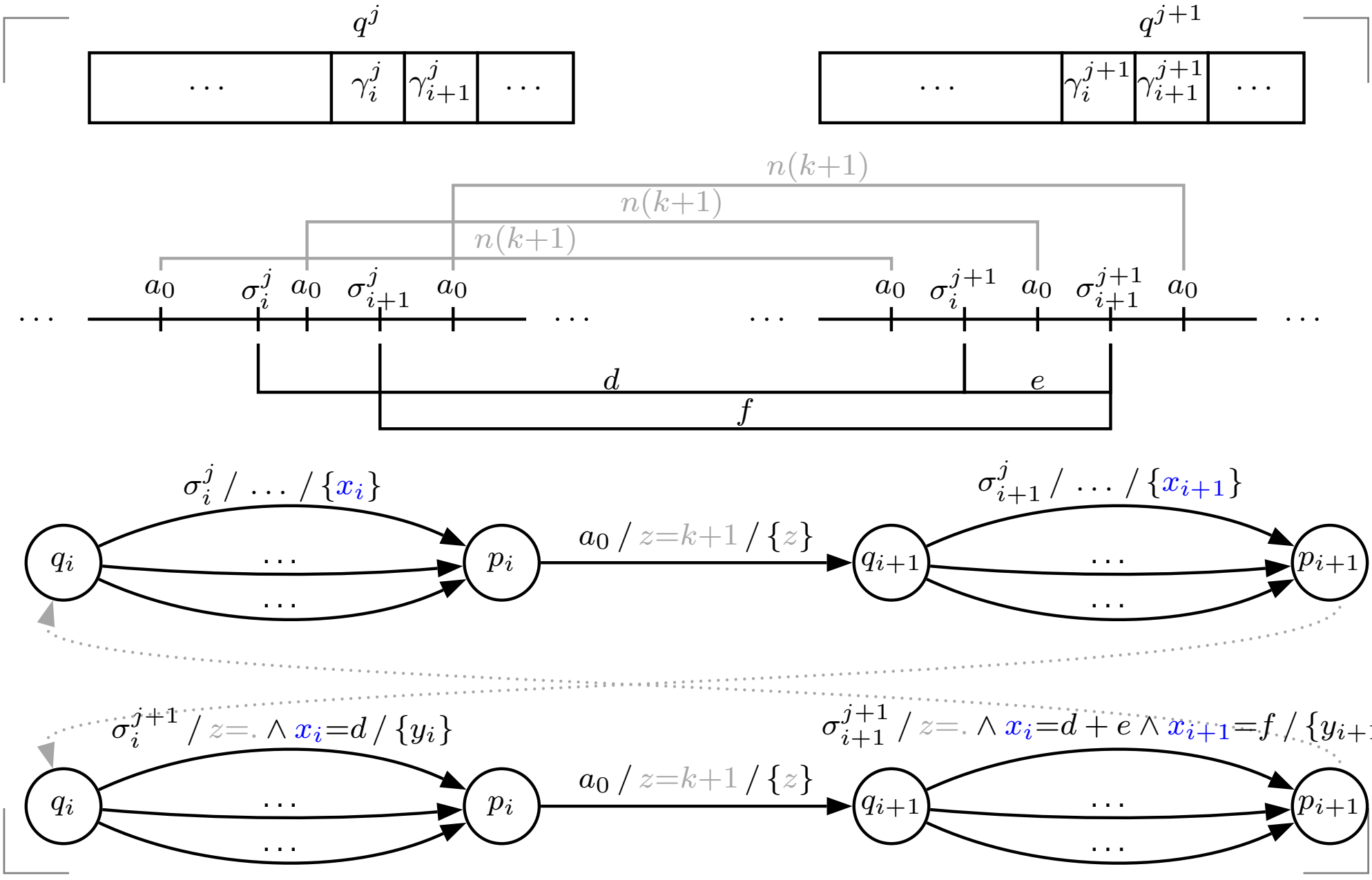
$Z = \{z\}$



Assurer le cohérence

- Début du chemin impose le codage de m
- 2 horloges pour chaque case
- Les unes sont lues pour connaître la configuration précédente
- Les autres sont remises à zéro pour la configuration suivante
- Contraintes sur les précédentes correspondant au le calcul de la nouvelle valeur

Exemple de transition



Complexité de la vacuité

- Le langage accepté n'est pas vide ssi M accepte m sans en sortir
- La taille de l'automate ainsi engendré est polynomial en $|M|$ et $|m|$
- CO-PSPACE = PSPACE (Savitch 70)
- Vacuité est donc PSPACE-complet
- De même sur les mots infinis [Alur and Dill, 1994]

Universalité... dans co-$\mathcal{R}.\mathcal{E}.$

Instance (Universalité)

$\mathcal{A}$: automate temporisé

Question

$\mathcal{A}$ accepte-t-il tous les mots ?

- Si un mot n'est pas accepté, il y a un mot à dates rationnelles non accepté
- Savoir si un mot à dates rationnelles est accepté est décidable
- Il y a un nombre dénombrable de mots à date rationnelles
- Universalité est donc dans co-$\mathcal{R}.\mathcal{E}.$

Universalité... complexité

co-$\mathcal{R}.\mathcal{E}.$-complet par co-réduction de

Instance ($\mathcal{R}.\mathcal{E}.$-complet car modèle universel [Minsky, 1967])

M : machine à deux compteurs (registres)

Question

M s'arrête-t-elle démarrée avec ses deux compteurs à zéro ?

Registre contenant des entiers positifs

Opérations sur les registres : $++$, $\div\div$ et

`si ..==0 aller ..`

Programme / code : suite d'opérations

Une configuration $< l, a, b >$

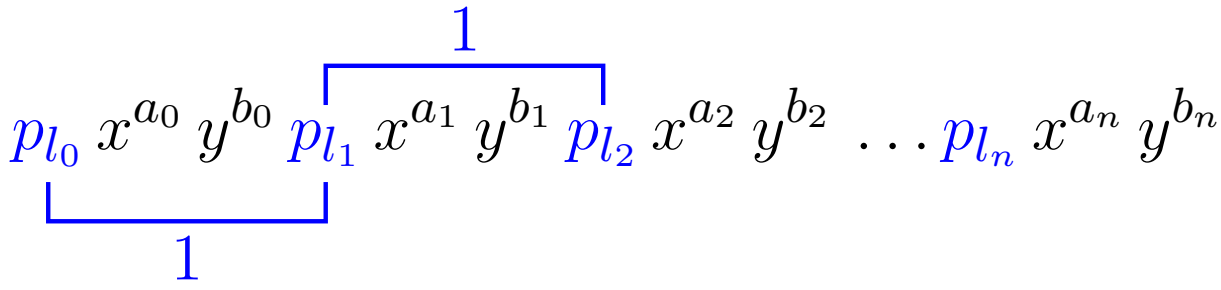
l : numéro de ligne

a et b valeurs des compteurs

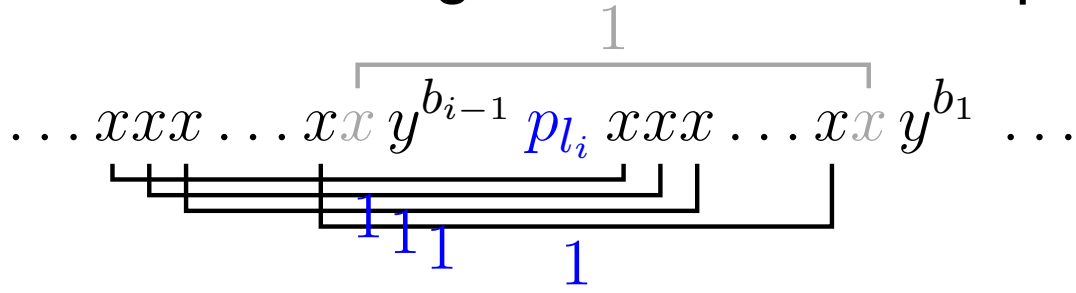
Codage des calculs

$$< l_0, a_0, b_0 > . < l_1, a_1, b_1 > . < l_2, a_2, b_2 > \ldots < l_n, a_n, b_n >$$

- Forme générale



- Nombre non borné de signaux dans un temps



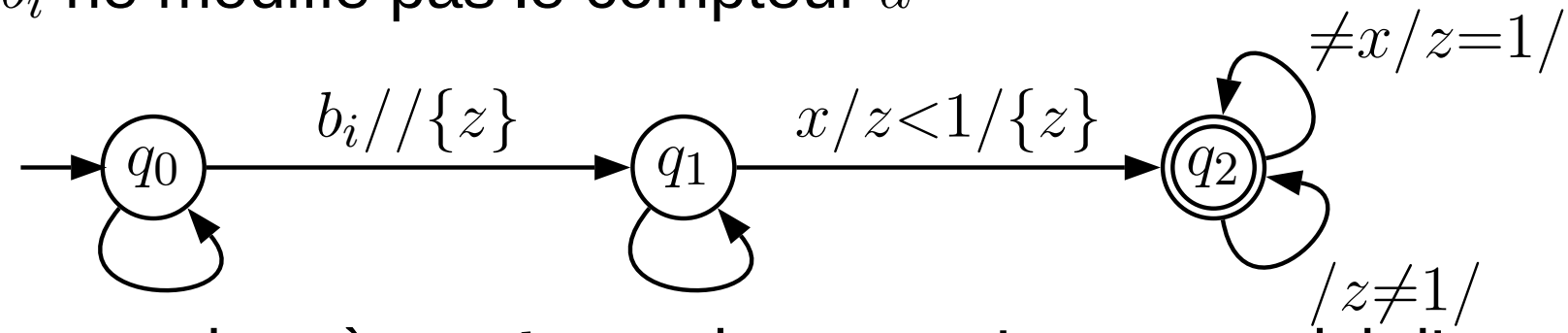
- De même pour les y

Réduction

- Seul mot *non accepté* correspond à un calcul s'arrêtant
- Par union d'automates (ce qui est clos)
  - exclu tout ce qui est de la forme $((p_1|p_2|p_3..p_m) \cdot x^* \cdot y^*)^*$
  - exclu si bon début ou bonne fin
  - exclu tout p_i en dehors des temps entiers
  - exclu tout temps entier sans p_i
- Traitement de chaque règle et correspondance de x et des y

Traitement des règles

- Si b_i ne modifie pas le compteur a



  accepte si après un b_i, un de ses x n'est pas suivi d'un autre x un temps après
- Adaptation quand il doit y avoir un x en plus ou en moins
- De même pour les y

- Mot accepté dès lors qu'une règle est violée
- Exclu par tous $\Rightarrow$ code un calcul s'arrêtant depuis $(0,0)$

Conséquences

- L'universalité est co-$\mathcal{R}.\mathcal{E}.$-complet
- Corollaire : $\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ est au moins co-$\mathcal{R}.\mathcal{E}.$-difficile
- $\mathcal{R}.\mathcal{E}.$ = Σ_1^0 et $co - \mathcal{R}.\mathcal{E}.$ = Π_1^0
  (hiérarchie arithmétique)
- Pour les mots temporisés infinis [Alur and Dill, 1994]
  l'universalité est Π_1^1-difficile et est dans Π_2^1
  (hiérarchie analytique *c.f.* [Rogers, 1967])
- La hiérarchie arithmétique quantifie sur des entiers
- La hiérarchie analytique quantifie sur des fonctions / ensembles (non nécessairement récursifs)

Conclusion

- Codage d'information dans le temps
  (dans les diagrammes espace-temps)
- Toujours une infinité de places entre 2 signaux
- Non clos par complément
- Les déterministes reconnaissent strictement moins de langages
- Divers manipulations ont de complexité / degrés de non-calculabilité très importantes

Extensions

- Entrées / sorties (transducteurs)
- Traces, commutation partielle (à définir)
- Traitement des accumulations (autrement qu'avec des ordinaux [Bérard and Picaronny, 2000])
- Version signal
- Littérature récente et relativement riche

Références

[Alur and Dill, 1994] Alur, R. and Dill, D. L. (1994). A Theory of timed automata. *Theoretical Computer Science*, 126(2):183–235.

[Asarin et al., 2002] Asarin, E., Caspi, P., and Maler, O. (2002). Timed regular expressions. *Journal of the ACM*, 49(2):172–206.

[Bérard and Picaronny, 2000] Bérard, B. and Picaronny, C. (2000). Accepting zeno words: a way towards timed refinements. *Acta Informatica*, 37(1):45–81.

[Hopcroft and Ullman, 1979] Hopcroft, J. and Ullman, J. (1979). *Introduction to Automata Theory. Languages, and Computation*. Addison-Wesley.

[Minsky, 1967] Minsky, M. (1967). *Finite and Infinite Machines*. Prentice Hall.

[Rogers, 1967] Rogers, H. (1967). *Theory of Recursive Functions and Effective Computability*. McGraw-Hill.