Άλγεβρες Διεργασιών (Process Algebras)

Στην ενότητα αυτή θα μελετηθούν τα εξής θέματα:

Προδιαγραφή και Επαλήθευση με άλγεβρες διεργασιών

Η άλγεβρα διεργασιών CCS

- *Ενέργειες και διεργασίες*
- *Σύνταξη*
- *Σημασιολογία*

Άλγεβρες Διεργασιών

- Μια εναλλακτική επιλογή για προδιαγραφή και επαλήθευση συστημάτων.
- Έμφαση δίνεται, σε "ανοικτά" συστήματα, δηλαδή σε συστήματα τα οποία δυνατόν να ενθυλακωθούν σε άλλα συστήματα.
- Μεγαλύτερη σημασία δίνεται στις *μεταβάσεις* παρά στις καταστάσεις ενός συστήματος (action-based vs state-based)
- Θεωρία που μελετά τις αλληλεπιδράσεις ενός συστήματος με το περιβάλλον του.
- Βασικός στόχος η *συνθετικότητα* (compositionality) δηλαδή η ικανότητα εξαγωγής συμπερασμάτων για ένα σύνθετο σύστημα από τις ιδιότητες των επιμέρους συνιστωσών του.

Άλγεβρες διεργασιών

- Περιλαμβάνουν
  - Μια *γλώσσα προδιαγραφής (μοντελοποίησης)* συστημάτων, που αποτελείται από ένα σύνολο τελεστών για σύνθεση συστημάτων σε μεγαλύτερα συστήματα.
  - *Αυστηρά διατυπωμένη σημασιολογία* που απεικονίζει τις δυνατές λειτουργίες και αλληλεπιδράσεις ενός συστήματος διατυπωμένου στη γλώσσα προδιαγραφής.
  - *Έννοιες εκλέπτυνσης συμπεριφοράς* που προσδιορίζουν πότε δύο συστήματα εμφανίζουν την ίδια συμπεριφορά, ή πότε ένα σύστημα "υλοποιεί" κάποιο άλλο.

Προδιαγραφή και Επαλήθευση σε ΑΔ

- Η ανάλυση ενός συστήματος με μία άλγεβρα διεργασιών συνήθως διενεργείται σε τρεις φάσεις:
  1. Διατύπωση της προδιαγραφής του συστήματος *Spec* (επιθυμητή συμπεριφορά και ιδιότητες) στη γλώσσα της ΑΔ
  2. Δημιουργία υποψήφιας υλοποίησης συστήματος *Imp*
  3. Έλεγχος κατά πόσο η υλοποίηση *Imp* ικανοποιεί την προδιαγραφή *Spec* δείχνοντας ότι την εκλεπτύνει/υλοποιεί.

- Σε ένα μοντέλο ΑΔ μπορούν επίσης να εκτελεσθούν μοντελο-έλεγχος και πειράματα.

Παράδειγμα 1

- Ένα ρολόι που κτυπά ασταμάτητα.

$$Clock \overset{def}{=} tick.Clock$$

– *tick*	το όνομα της ενέργειας
– *Clock*	το όνομα της διεργασίας
$\overset{def}{=}$	συνδέει ένα όνομα διεργασίας με την έκφραση μιας διεργασίας
– *tick.Clock*	η διεργασία
– '.'	τελεστής αλληλουχίας

Συμπεριφορά και μεταβάσεις

- Η συμπεριφορά μιας διεργασίας συλλαμβάνεται μέσω μεταβάσεων. Αν E και F είναι δύο διεργασίες και a μια ενέργεια, τότε γράφουμε

$$E \xrightarrow{a} F$$

για να δείξουμε ότι η διεργασία E είναι ικανή να εκτελέσει την ενέργεια a και να εξελιχθεί στη διεργασία F.

- Υπάρχει ένα σύνολο από κανόνες (η *σημασιολογία* της ΑΔ) που καθορίζουν τη συμπεριφορά των διεργασιών μιας άλγεβρας διεργασιών.

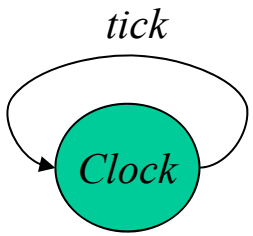
- Παράδειγμα:

$$Clock \xrightarrow{tick} Clock$$

Συμπεριφορά και συστήματα μεταβάσεων

- Μέσω των κανόνων της σημασιολογίας μιας ΑΔ η συμπεριφορά κάθε διεργασίας μεταφράζεται σε ένα γράφο με βάρη (σύστημα μεταβάσεων), όπου
  - Οι κορυφές του γράφου αντιστοιχούν σε διεργασίες
  - Οι ακμές συλλαμβάνουν τις μεταβάσεις των διεργασιών
  - Τα βάρη αναφέρονται στα ονόματα των εκτελεσθέντων ενεργειών

- Παράδειγμα:



Άλγεβρες Διεργασιών - Σύνοψη

Σύνολο από τελεστές.

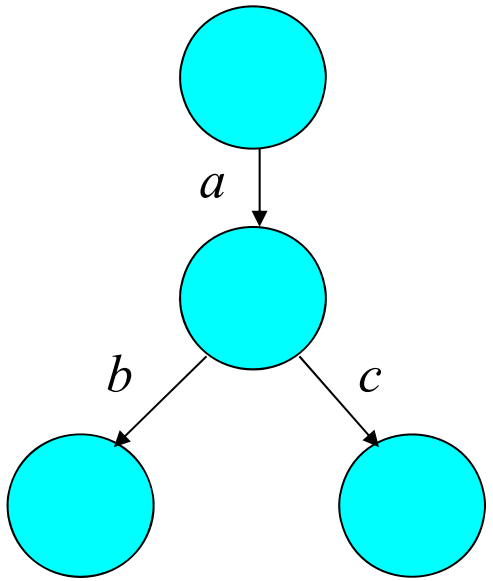
- Συνήθεις τελεστές
  - 0 (NULL) αδιέξοδο/τερματισμός.
  - *a*.E – εκτέλεσε την ενέργεια *a*, και συνέχισε σύμφωνα με το E .
  - E+F – συνέχισε σύμφωνα με το E ή το F.
  - E||F – εκτέλεσε τα E και F παράλληλα

Σημασιολογία

- Συνήθης μέθοδος:
  - σύνολο από κανόνες που συλλαμβάνουν τη συμπεριφορά κάθε ενός από τους τελεστές.
  - Με βάση τους κανόνες η συμπεριφορά του συστήματος ορίζεται ως ένας γράφος.

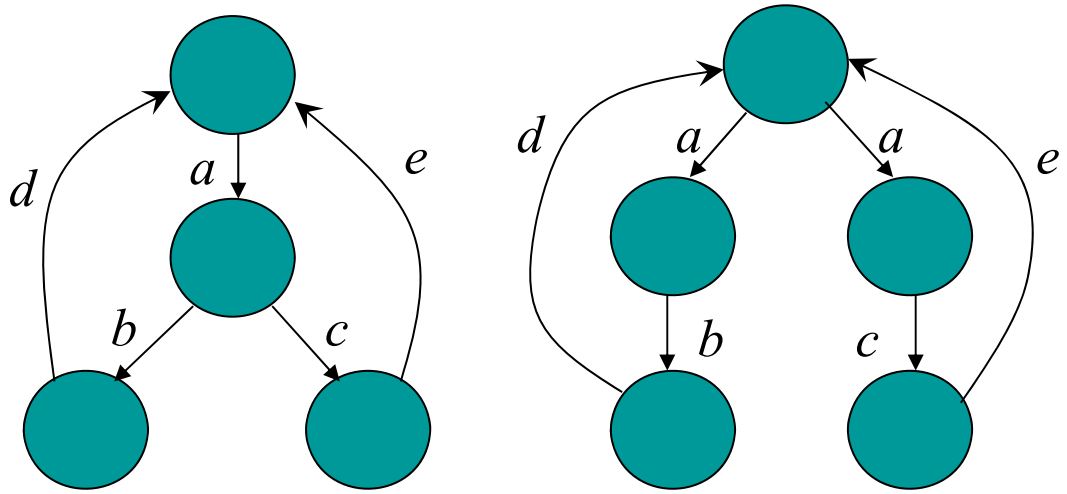
Παράδειγμα:

$a.(b+c)$ ή $a.(b.0+c.0)$



Μοντέλα και εκτελέσεις

Εκτέλεση μίας διεργασίας είναι οποιοδήποτε μονοπάτι μπορεί να ληφθεί από τον γράφο της διεργασίας.



Μονοπάτια: *abdabdabdaceabdabd*...

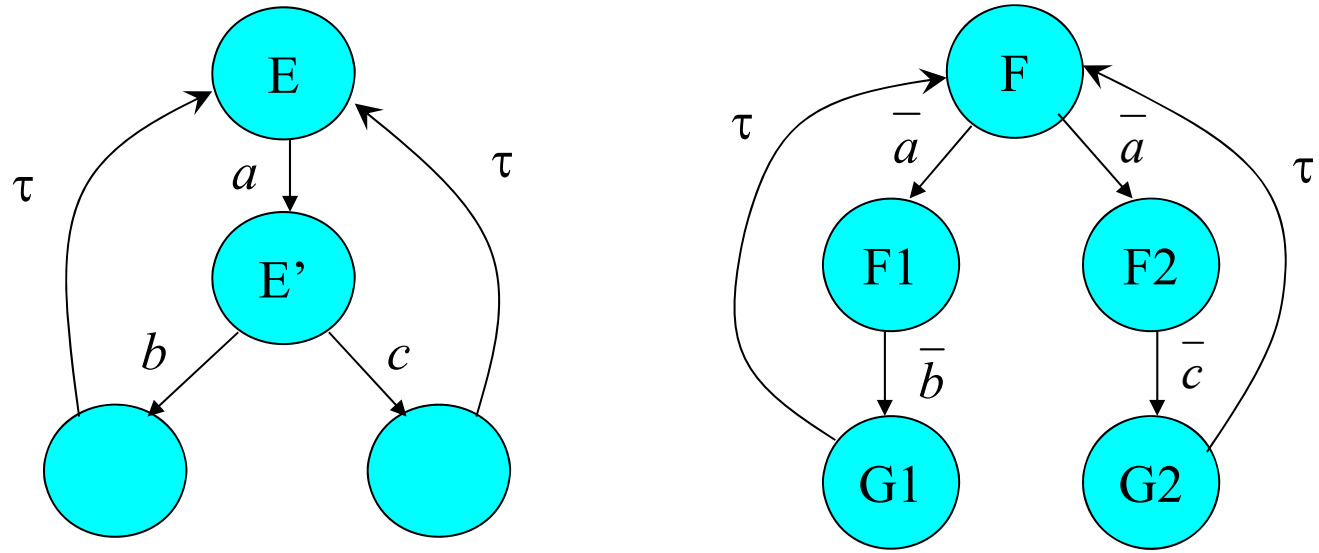
a: insert coin, *b*: press pepsi, *c*: press pepsi-light

d: get pepsi, *e*: get pepsi-light

Σχέσεις ισοδυναμίας μας επιτρέπουν να καθορίσουμε πότε θεωρούμε δύο συστήματα/ διεργασίες ταυτόσημα.

Τα δύο συστήματα έχουν τις ίδιες εκτελέσεις, είναι όμως *διαφορετικά*...!

Ενέργειες και αλληλενέργιες



Για κάθε ενέργεια *α*, υπάρχει μια αλληλενέργεια $\bar{a}$.

Οι δύο ενέργειες απεικονίζουν *είσοδο* και *έξοδο* αντίστοιχα και προσφέρουν τη δυνατότητα αλληλεπίδρασης σε άλγεβρες διεργασιών.

CCS: Calculus of Communicating Systems

- Θα μελετήσουμε τη χρήση αλγεβρών διαδικασιών για μοντελοποίηση/ανάλυση συστημάτων μέσω της άλγεβρας διεργασιών **CCS.**

- Αναπτύχθηκε από τον Robin Milner (Turing Award, 1991) κατά τα τέλη του 1970/αρχές 1980.

- Παρουσιάζει τον *δυαδικό συγχρονισμό* ως βασική μέθοδο επικοινωνίας.

- Οι διεργασίες κτίζονται από ένα σύνολο ατομικών ενεργειών με τη χρήση τελεστών σύνθεσης διεργασιών.

Ενέργειες στη CCS

- Μπορούν να είναι *ενέργειες εισόδου*, *ενέργειες εξόδου*, ή η *εσωτερική ενέργεια*.
- Έστω σύνολο από *ετικέτες* (ονόματα καναλιών) Λ και η ετικέτα τ. Μια ενέργεια στη CCS μπορεί να είναι ένα από τα ακόλουθα
  - ενέργεια εισόδου στο κανάλι $\lambda \in \Lambda$: λ
  - ενέργεια εξόδου στο κανάλι $\lambda \in \Lambda$: $\overline{\lambda}$
  - η εσωτερική ενέργεια: τ
- Συμβολισμός
  - Λ το σύνολο ετικετών και το σύνολο των ενεργειών εισόδου
  - $\overline{\Lambda}$ το σύνολο των ενεργειών εξόδου
  - $\Lambda \cup \overline{\Lambda}$ το σύνολο των εξωτερικών ενεργειών
  - $Act = \Lambda \cup \overline{\Lambda} \cup \tau$ το σύνολο όλων των ενεργειών

Ενέργειες και αλληλενέργειες

- Σε άλγεβρες διεργασιών όπως η CCS συστήματα επικοινωνούν με το περιβάλλον τους (και μεταξύ τους) μέσω συγχρονισμών που συμβαίνουν σε κανάλια.
  - Αν μια διεργασία είναι διαθέσιμη για είσοδο και κάποια άλλη για έξοδο στο ίδιο κανάλι, τότε ο συγχρονισμός λαμβάνει χώρο και οι διεργασίες προχωρούν με την εκτέλεσή τους.
  - Οι εξωτερικές ενέργειες που μπορεί να εκτελέσει ένα σύστημα μπορούν να θεωρηθούν ως η συμπεριφορά/διεπιφάνειά του.

Η σύνταξη της CCS

- Έστω $\alpha \in Act$, $L \subseteq \Lambda$, $f : \Lambda \to \Lambda$, $C \in \mathbf{C}$ όπου **C**, ένα σύνολο από *ονόματα διεργασιών*. Η πιο κάτω γραμματική δίνει τη σύνταξη των διεργασιών της CCS:

E	$::=$	0	τερματισμός
	$\mid$	$a.E$	διαδοχή
	$\mid$	$E_1 + E_2$	επιλογή
	$\mid$	$E_1 \mid E_2$	ταυτοχρονισμός
	$\mid$	$E \backslash L$	περιορισμός
	$\mid$	$E[f]$	μετονομασία
	$\mid$	C	κλήση

- Ο τελεστής '.' έχει μεγαλύτερη προτεραιότητα από τον '+', ενώ η κατάληξη '.0' και όροι όπως '… | 0' συχνά παραλείπονται.

CCS τελεστές και άτυπη ερμηνεία

- Με βάση την πιο πάνω περιγραφή μπορούμε να θεωρήσουμε τους τελεστές της CCS ως πράξεις για κτίσιμο και συνδυασμό κουτιών ως εξής:

0	:	Διεργασία που δεν ανταποκρίνεται (τερματισμός ή αδιέξοδο)
$a.E$	:	Διεργασία πρόθυμη να εκτελέσει την ενέργεια α και στη συνέχεια συμπεριφέρεται ως E
$E_1 + E_2$	:	Διεργασία η οποία προσφέρει την επιλογή ανάμεσα στις E_1 και E_2

CCS τελεστές και άτυπη ερμηνεία

$E_1 \mid E_2$: Διεργασία στην οποία τρέχουν παράλληλα οι E_1 και E_2. Οι υποδιεργασίες αυτές μπορούν να εκτελούν ανεξάρτητα την εργασία τους ή και να επικοινωνούν μεταξύ τους στα κανάλια που τις συνδέουν.

$E \setminus L$: Διεργασία με τα κανάλια L δεσμευμένα για χρήση αποκλειστικά εντός της E.

$E[f]$: Η διεργασία E αφού έχουν μετονομασθεί τα κανάλια της σύμφωνα με το f

Η σύνταξη της CCS

- Μία CCS-έκφραση E ονομάζεται *κλειστή* (closed) αν κάθε όνομα διεργασίας που περιέχεται σε αυτή έχει δηλωθεί.

- Δηλώσεις έχουν τη μορφή: $C \overset{def}{=} E$.

- Παράδειγμα: Έστω ότι δηλώνουμε το όνομα διεργασίας A ως

  $$A \overset{def}{=} a.b.A$$

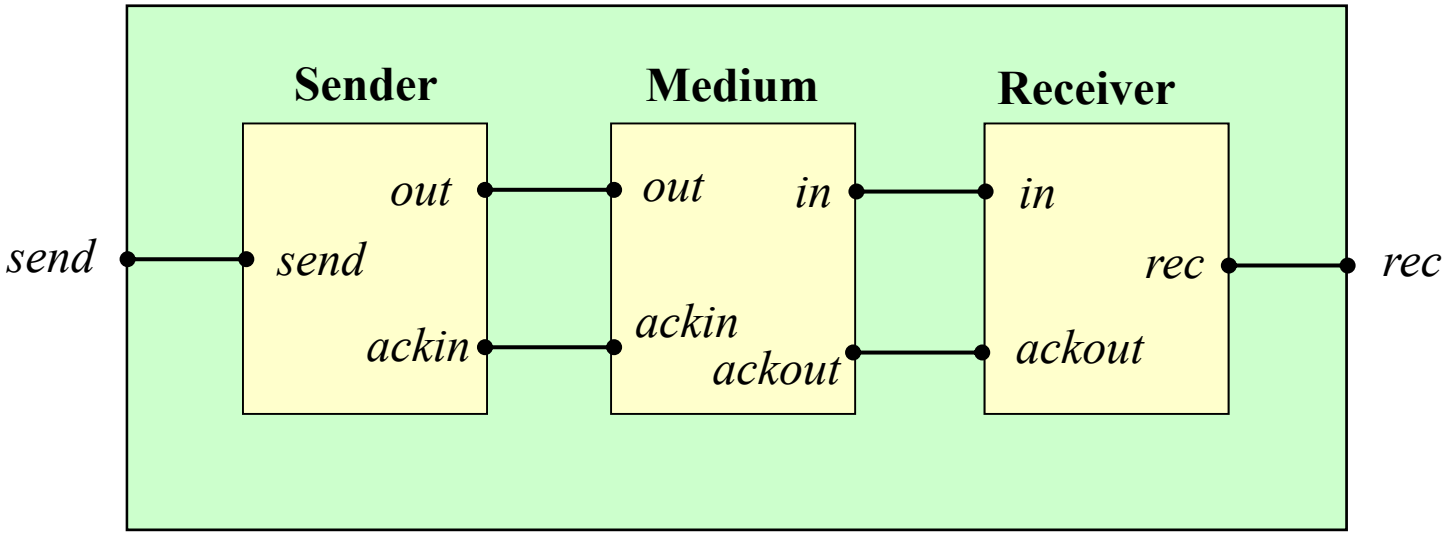
  τότε οι όροι $A \mid A,\ c.A$ είναι κλειστοί.

- *Διεργασίες της CCS* ονομάζουμε όλες τις κλειστές CCS εκφράσεις.

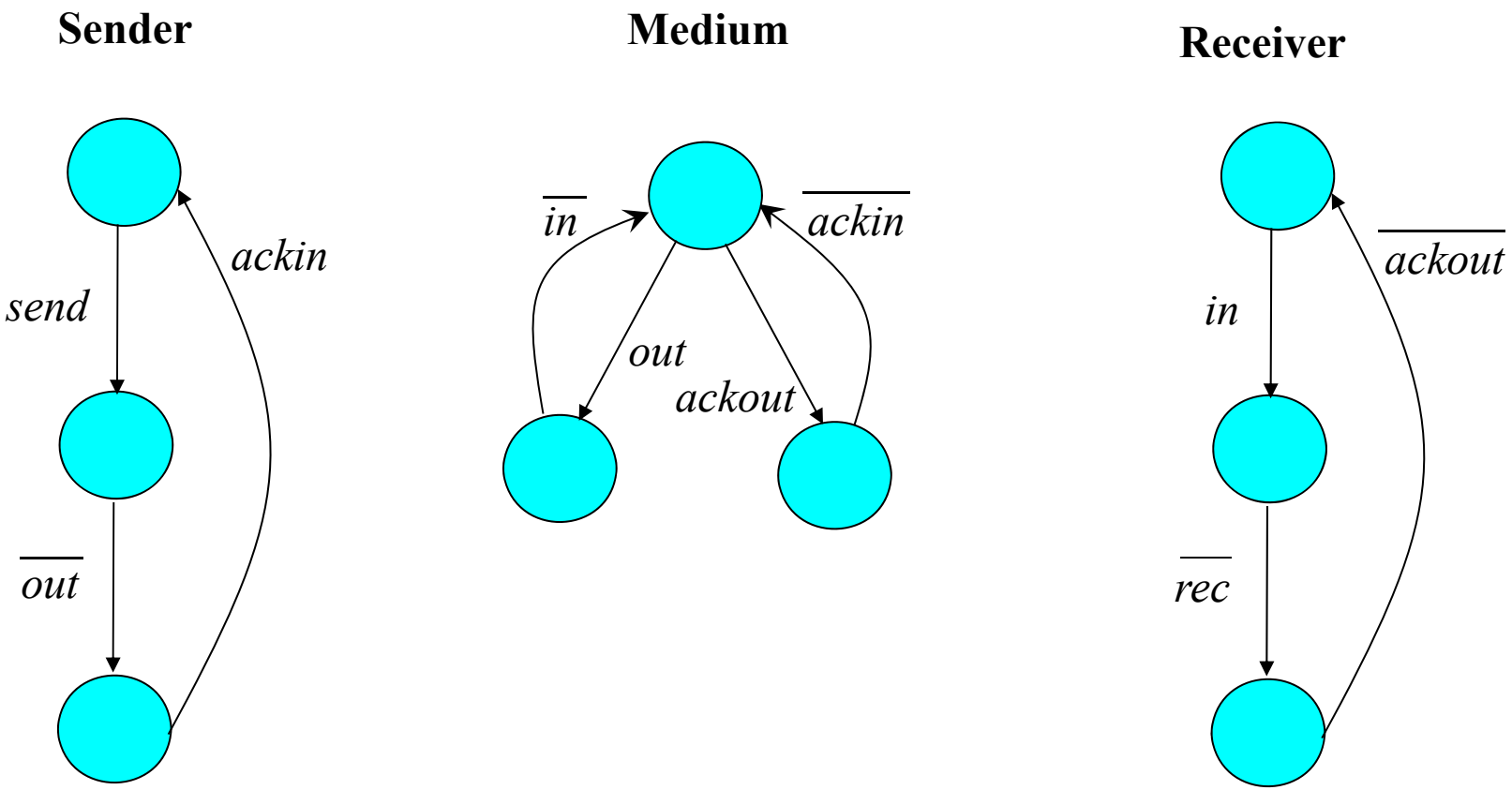
- Γράφουμε **Proc** για το σύνολο των CCS διεργασιών.

Μοντελοποίηση

- Με τη CCS είναι δυνατή η περιγραφή συστημάτων όπως το πιο κάτω ως η σύνθεση των υποκομματιών του.



Τα συστατικά του συστήματος



Τα συστατικά του συστήματος

$$Sender = send.\ \overline{out}.\ ackin.\ Sender$$

$$Medium = out.\ \overline{in}.\ Medium + ackout.\ \overline{ackin}.\ Medium$$

$$Receiver = in.\ \overline{rec}.\ \overline{ackout}.\ Receiver$$

$$System = (\ Sender\ |\ Medium\ |\ Receiver\ )\setminus \{\ in,\ out,\ ackin,\ ackout\}$$

Άσκηση

- Θεωρείστε ένα εργαστήριο στο οποίο αντικείμενα φθάνουν μέσω μιας ζώνης σε κάποιο χώρο όπου τυγχάνουν επεξεργασίας με τη χρήση ενός εργαλείου. Όταν η επεξεργασία τελειώσει το αντικείμενο τοποθετείται σε μία δεύτερη ζώνη από όπου απομακρύνεται από το εργαστήριο. Στο εργαστήριο υπάρχουν δύο άτομα που ασχολούνται με την συγκεκριμένη εργασία και τα οποία μοιράζονται τη χρήση ενός εργαλείου.

- Να ορίσετε το σύστημα στη CCS χρησιμοποιώντας (ανάμεσα σε άλλα) τα κανάλια in και out, μέσω των οποίων καταφθάνουν και απομακρύνονται τα αντικείμενα.

Ερμηνεία των CCS τελεστών

- Έχουμε δει τους τελεστές της CCS και άτυπα την επιδιωκόμενη συμπεριφορά τους.
- Επόμενο βήμα είναι η τυπική διατύπωση της συμπεριφοράς των CCS τελεστών μέσω μιας σημασιολογίας. Η σημασιολογία που θα δώσουμε είναι λειτουργική (operational semantics), δηλαδή περιγράφει τη λειτουργία μιας CCS διεργασίας.
  - Θα ορίζει τα βήματα εκτέλεσης της CCS μεταφράζοντας μια διεργασία σε ένα σύστημα μεταβάσεων
  - Η σημασιολογία θα δίνει το πλαίσιο στο οποίο θα οριστούν μεθοδολογίες για ανάλυση CCS μοντέλων

Σημασιολογία της CCS

- Η σημασιολογία της CCS ορίζεται μαθηματικά ως η σχέση

$$\rightarrow \subseteq \mathbf{Proc} \times Act \times \mathbf{Proc}$$

όπου

  - $\langle E, a, F\rangle \in \rightarrow$ σημαίνει πως η διεργασία E μπορεί να εκτελέσει την ενέργεια a και μετά να εξελιχθεί στη διεργασία F.

  - Γράφουμε $E \xrightarrow{a} F$ αντί του $\langle E, a, F\rangle \in \rightarrow$.

Σημασιολογία της CCS

- Η σχέση $\rightarrow$ ορίζεται αναδρομικά ως ένα σύνολο από κανόνες της μορφής

$$\frac{\text{συνθήκες μεταβάσεων}}{\text{συμπέρασμα}} \quad \text{επιμέρους συνθήκη}$$

- Οι κανόνες δηλώνουν πως αν ισχύουν οι συνθήκες μεταβάσεων, οι οποίες έχουν τη μορφή $E \xrightarrow{a} F$, και η επιμέρους συνθήκη, τότε ισχύει το συμπέρασμα (η ζητούμενη μετάβαση).

- Για κάθε τελεστή της CCS υπάρχει μία ομάδα κανόνων. Για αυτό το λόγο η σημασιολογία ονομάζεται Δομική Λειτουργική Σημασιολογία (Structured Operational Semantics – SOS).

SOS κανόνες I

Act $\dfrac{-}{a.E \xrightarrow{a} E}$

Sum$_1$ $\dfrac{E \xrightarrow{a} E'}{E + F \xrightarrow{a} E'}$

Sum$_2$ $\dfrac{F \xrightarrow{a} F'}{E + F \xrightarrow{a} F'}$

SOS κανόνες II

$$\textbf{Com}_1 \quad \frac{E \xrightarrow{a} E'}{E \mid F \xrightarrow{a} E' \mid F}$$

$$\textbf{Com}_2 \quad \frac{F \xrightarrow{a} F'}{E \mid F \xrightarrow{a} E \mid F'}$$

$$\textbf{Com}_3 \quad \frac{E \xrightarrow{a} E', F \xrightarrow{\bar{a}} F'}{E \mid F \xrightarrow{\tau} E' \mid F'}$$

SOS κανόνες III

Res $$\frac{E \xrightarrow{a} E'}{E \setminus L \xrightarrow{a} E' \setminus L} \quad a, \bar{a} \notin L$$

Rel $$\frac{E \xrightarrow{a} E'}{E[f] \xrightarrow{\hat{f}(a)} E'[f]}$$

Con $$\frac{E \xrightarrow{a} E'}{C \xrightarrow{a} E'} \quad C \in \mathrm{C},\ C \stackrel{def}{=} E$$

Σχόλια για τους κανόνες

- Ο κανόνας **Act** δεν περιέχει συνθήκες επομένως μπορεί να θεωρηθεί ως αξίωμα.
- Το αποτέλεσμα του συγχρονισμού που περιγράφεται στον κανόνα **Com$_3$** είναι η εσωτερική ενέργεια τ.
- Δεν υπάρχει κανόνας (δυνατή μετάβαση) για τη διεργασία τερματισμού *0*.
- Για $f : \Lambda \to \Lambda$, ορίζουμε $\hat{f} : Act \to Act$ ως εξής:

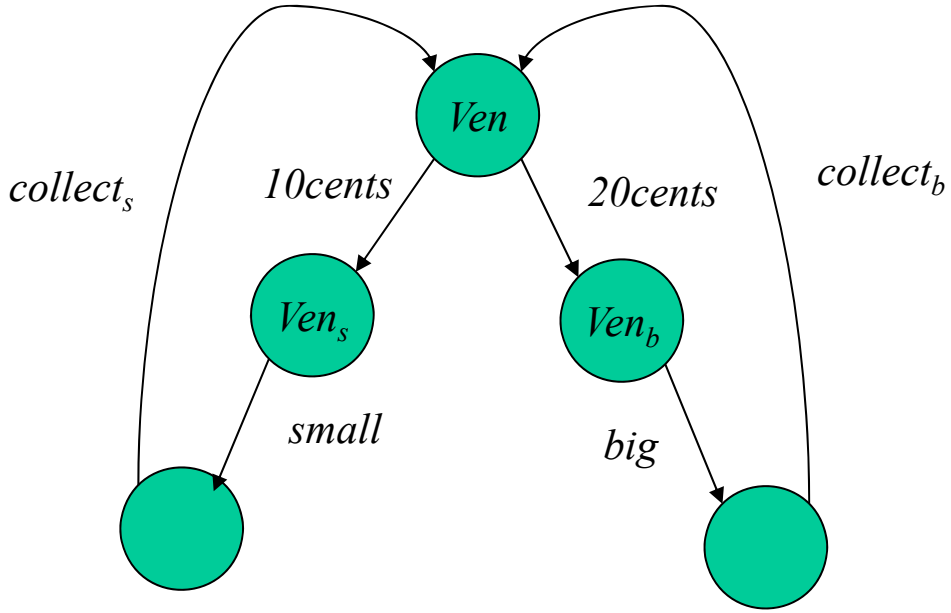
$$\hat{f}(a) = \begin{cases} f(a) & \text{αν } a \in \Lambda \\ \overline{f(b)} & \text{αν } a = \bar{b}, b \in \Lambda \\ \tau & \text{αν } a = \tau \end{cases}$$

Παράδειγμα – ο τελεστής +

$$Ven \overset{def}{=} 10cents.Ven_s + 20cents.Ven_b$$

$$Ven_s \overset{def}{=} small.collect_s.Ven$$

$$Ven_b \overset{def}{=} big.collect_b.Ven$$



Παράδειγμα – ο τελεστής |

$$Ven \overset{def}{=} 10cents.Ven_s + 20cents.Ven_b$$

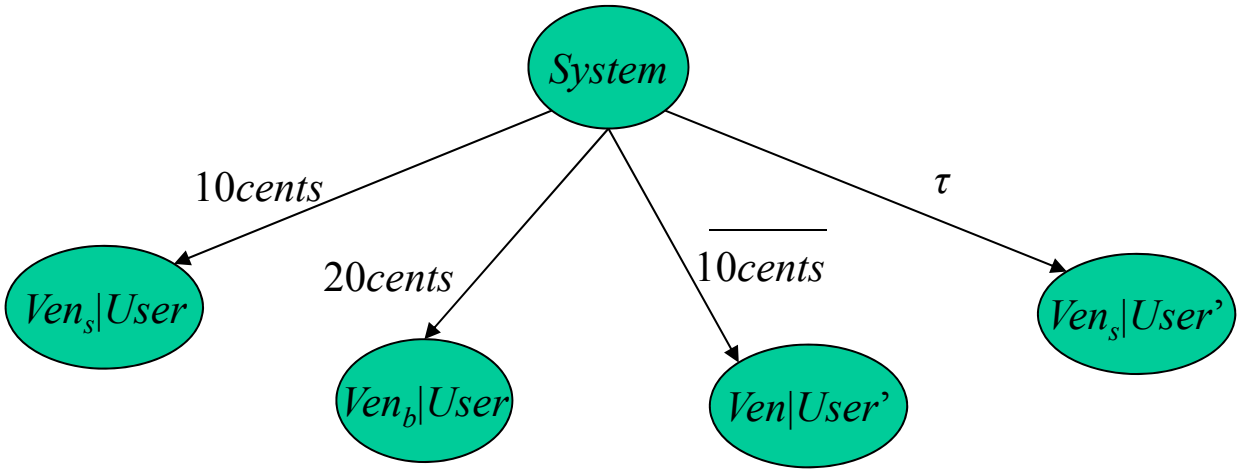
$$Ven_s \overset{def}{=} small.collect_s.Ven$$

$$Ven_b \overset{def}{=} big.collect_b.Ven$$

$$User \overset{def}{=} \overline{10cents}.User'$$

$$User' \overset{def}{=} \overline{small}.\overline{collect_s}.0$$

$$System \overset{def}{=} Ven \mid User$$



Παράδειγμα – ο τελεστής \

$$Ven \overset{def}{=} 10cents.Ven_s + 20cents.Ven_b$$

$$Ven_s \overset{def}{=} small.collect_s.Ven$$

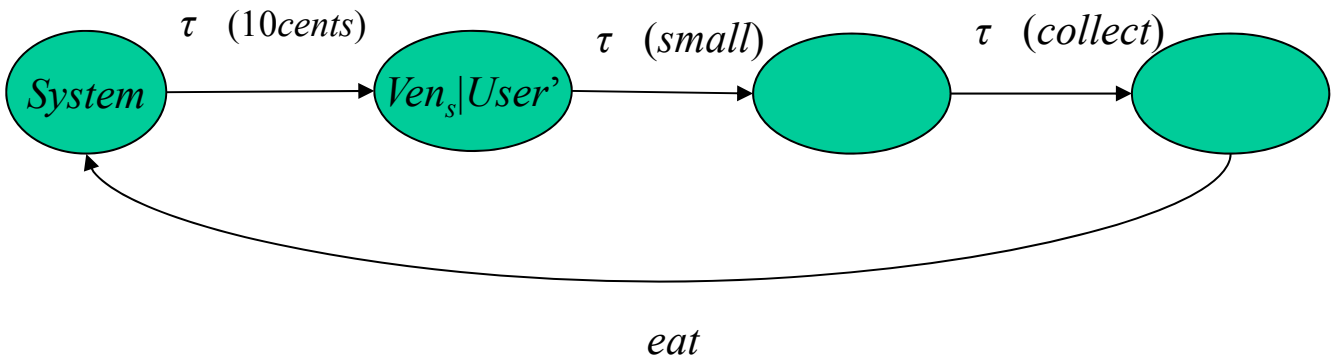
$$Ven_b \overset{def}{=} big.collect_b.Ven$$

$$User \overset{def}{=} \overline{10cents}.User'$$

$$User' \overset{def}{=} \overline{small}.\overline{collect_s}.eat.User$$

$$System \overset{def}{=} (Ven \mid User) \setminus L$$

$$L = \{10cents, 20cents, small, big, collect_s, collect_b\}$$



Παραδείγματα

$$a.(b.(c.0 \mid \bar{c}.0) + d.0) \xrightarrow{a} b.(c.0 \mid \bar{c}.0) + d.0$$

$$b.(c.0 \mid \bar{c}.0) + d.0 \xrightarrow{b} c.0 \mid \bar{c}.0 \qquad b.(c.0 \mid \bar{c}.0) + d.0 \xrightarrow{d} 0$$

$$c.0 \mid \bar{c}.0 \xrightarrow{c} \bar{c}.0 \qquad c.0 \mid \bar{c}.0 \xrightarrow{\bar{c}} c.0 \qquad c.0 \mid \bar{c}.0 \xrightarrow{\tau} 0 \mid 0$$

$$c.0 \xrightarrow{c} 0 \qquad \bar{c}.0 \xrightarrow{\bar{c}} 0 \qquad d.0 \xrightarrow{d} 0$$

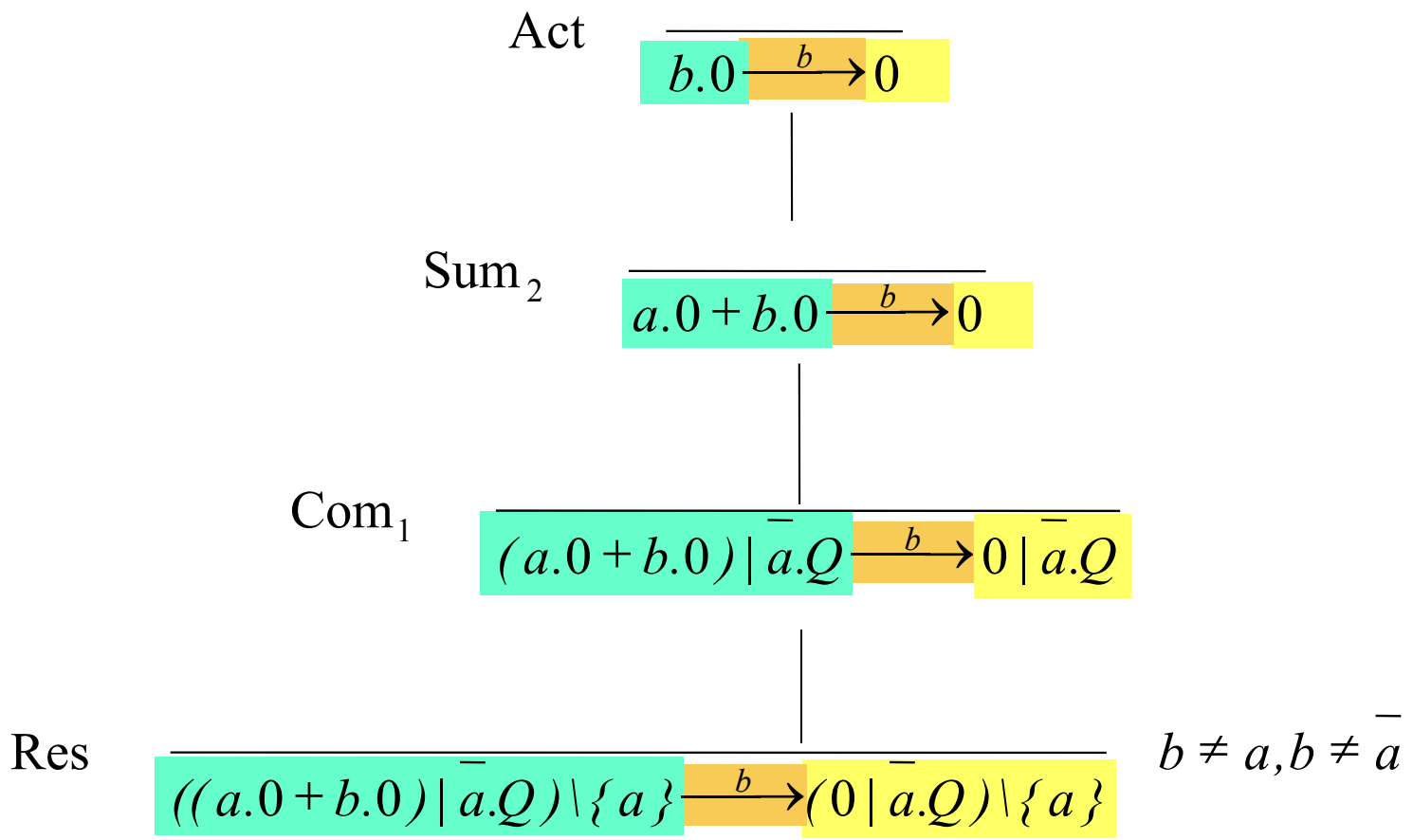
$$((c.0 \mid \bar{c}.0) + d.0) \setminus \{c,d\} \xrightarrow{\tau} 0 \mid 0$$

Κανόνες SOS και μεταβάσεις CCS διεργασιών

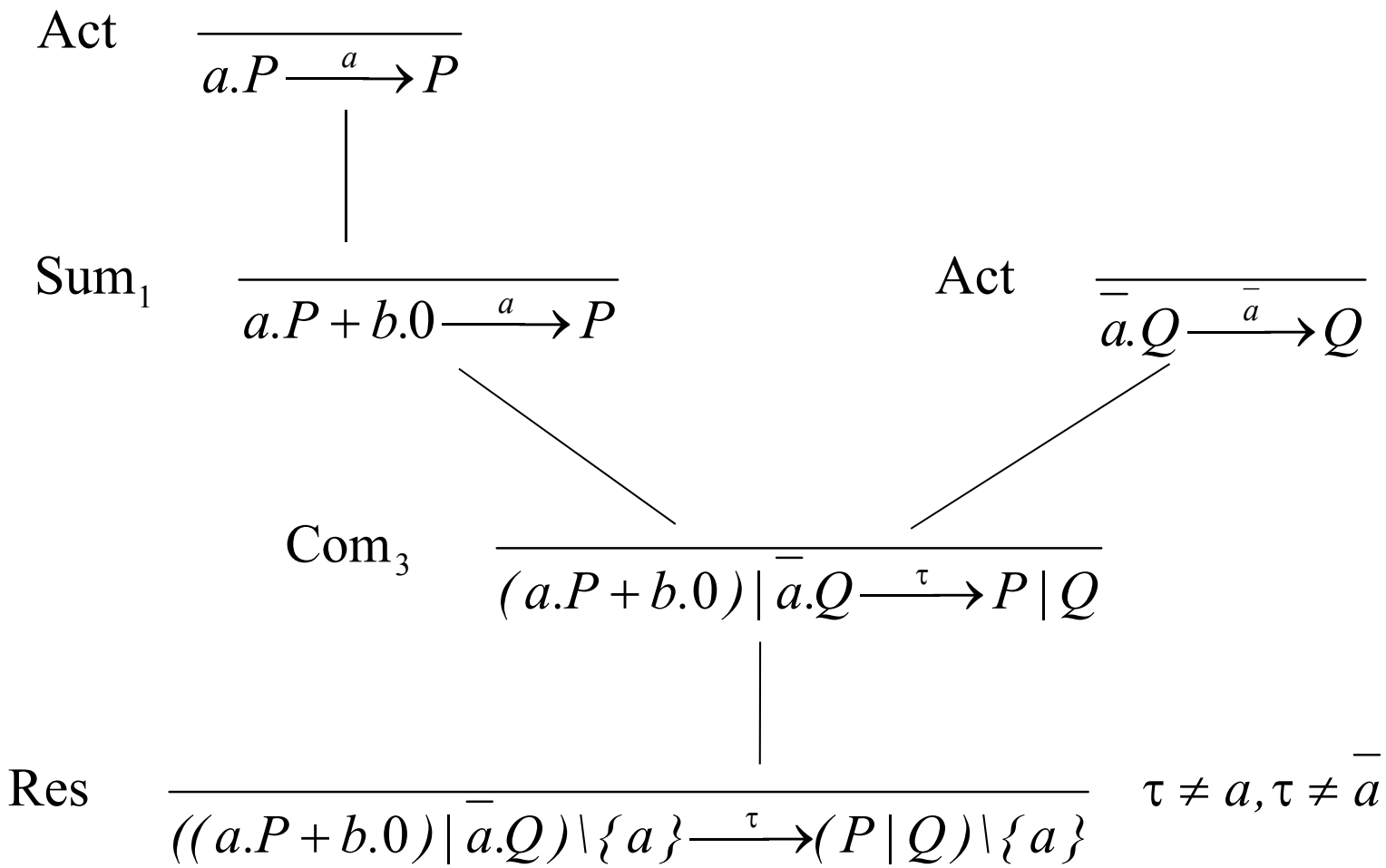
- Οι SOS κανόνες ορίζουν ένα σύστημα παραγωγής μεταβάσεων, όπου μια μετάβαση της μορφής $E \xrightarrow{a} F$ θεωρείται αληθής αν μπορεί να κτιστεί μια απόδειξη της μετάβασης χρησιμοποιώντας τους κανόνες.

- Επομένως η σχέση $\rightarrow$ περιέχει ακριβώς εκείνες τις μεταβάσεις που μπορούν να αποδειχθούν μέσω των κανόνων.

Παράδειγμα 1



Παράδειγμα 2



Αποδείξεις και τελεστές

- Οι αποδείξεις κτίζονται αλυσιδωτά, ξεκινώντας από κάποια/ες εφαρμογή/ες του αξιώματος **Act**.

- Η επιμέρους συνθήκη πρέπει να ισχύει σε κάθε εφαρμογή του κανόνα **Res**. Για παράδειγμα η μετάβαση

$$((a.P + b.0) \mid \overline{a}.Q) \setminus \{a\} \xrightarrow{a} (P \mid \overline{a}.Q) \setminus \{a\}$$

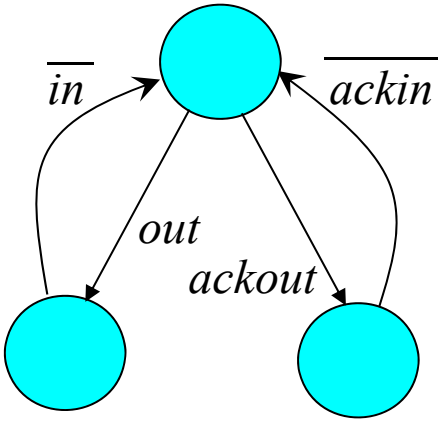
  δεν μπορεί να αποδειχθεί.

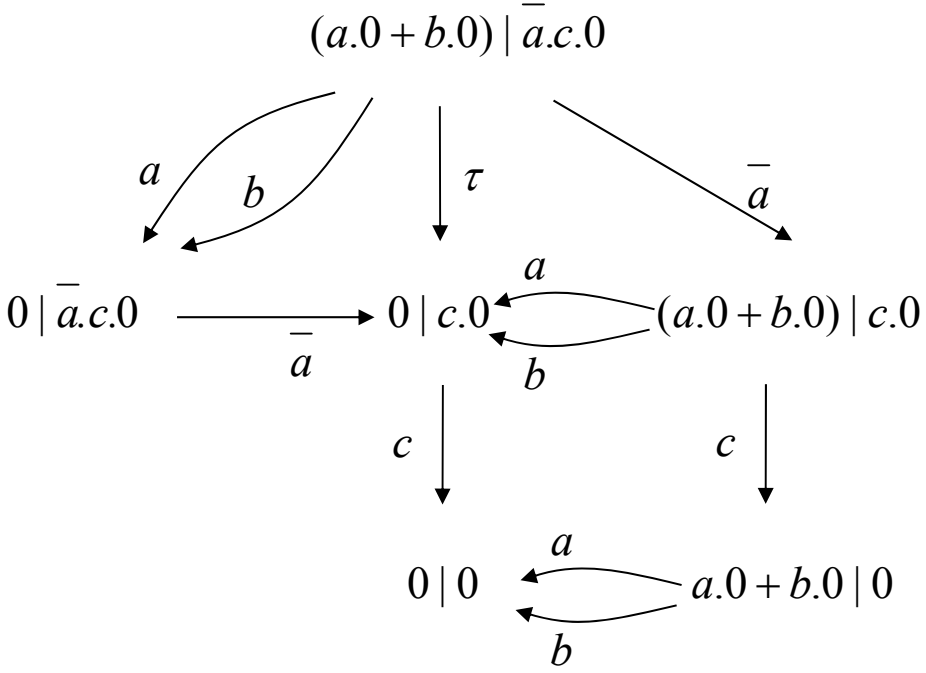
- Με βάση τους κανόνες οι τελεστές της CCS μπορούν να χωριστούν σε δύο κατηγορίες:
  - Δυναμικοί: *0, a., +*
  - Στατικοί: |, \L, [*f*]

  Βασική διαφορά: κατά πόσο διατηρούνται μετά από τους κανόνες μεταβάσεων.

Συστήματα μεταβάσεων με ετικέτες

- Σε κάθε διεργασία της CCS αντιστοιχεί κάποιο *σύστημα μεταβάσεων με ετικέτες* (labeled transition system, LTS) που περιγράφει τις δυνατές συμπεριφορές της διεργασίας.





CCS και LTSs

- Μπορούμε να θεωρήσουμε τη CCS ως ένα LTS (με μη πεπερασμένο αριθμό καταστάσεων) χωρίς αρχική κατάσταση, όπου
  - οι καταστάσεις είναι κλειστές διεργασίες
  - οι μεταβάσεις δίνονται από τη σημασιολογία

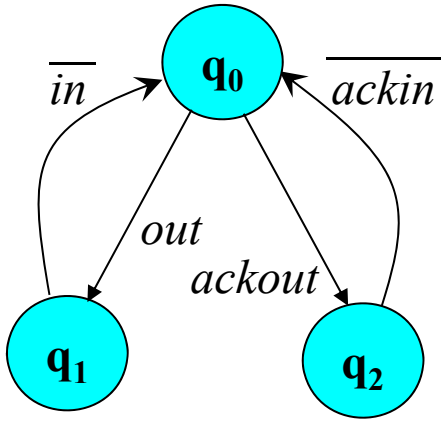
- Οποιοδήποτε LTS μπορεί να γραφτεί ως μια CCS διεργασία
  - Συνδέουμε ένα όνομα διεργασίας S με κάθε κατάσταση του LTS s.
  - Στη δήλωση της διεργασίας S αθροίζουμε τους όρους $a.T$ για κάθε μετάβαση $s \xrightarrow{a} t$ του συστήματος μεταβάσεων.

- Η κωδικοποίηση χρησιμοποιεί μόνο δυναμικούς τελεστές και δηλώσεις.

- Και οι στατικοί τελεστές; Χρησιμοποιούνται για την κωδικοποίηση της "αρχιτεκτονικής" ενός συστήματος.

Παράδειγμα

- Για το LTS



οι CCS δηλώσεις είναι

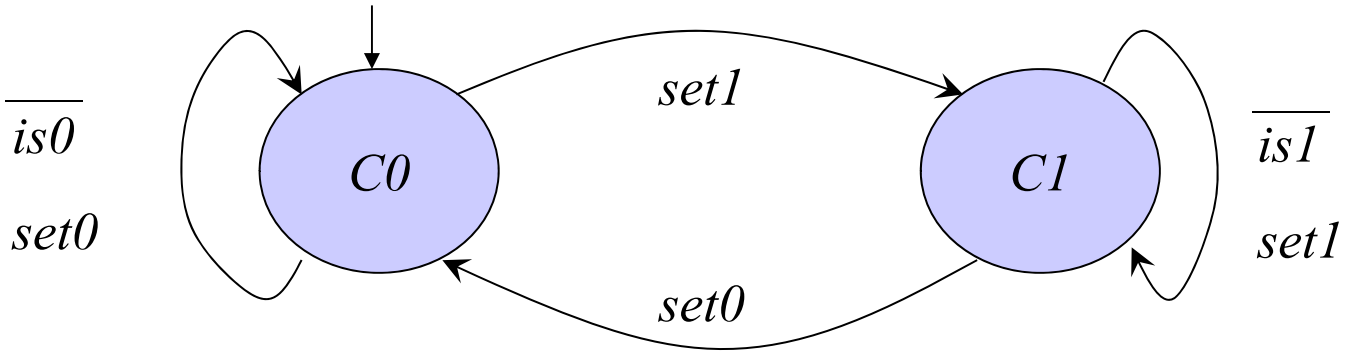
$$Q_0 = out.Q_1 + ackout.Q_2$$

$$Q_1 = \overline{in}.Q_0$$

$$Q_2 = \overline{ackin}.Q_0$$

και η διεργασία που περιγράφει τη συμπεριφορά του συστήματος από την αρχική κατάσταση είναι η Q_0.

Μοντελοποίηση μεταβλητής τύπου bit

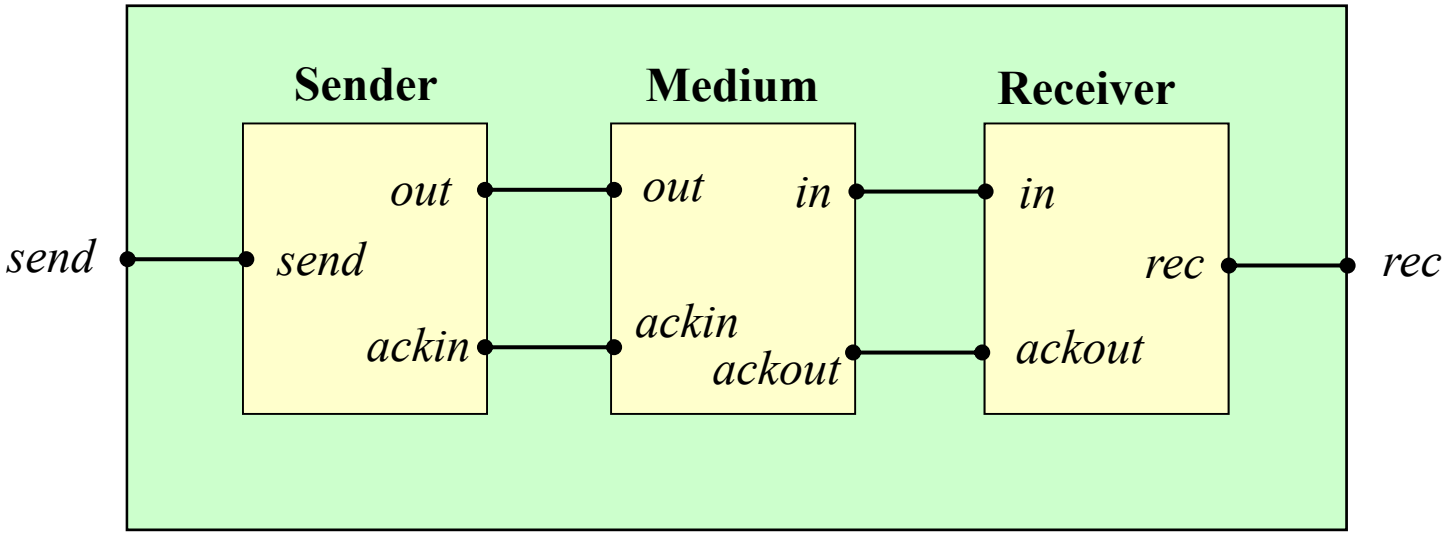


$$C0 = \overline{is0}.\ C0 \ + \ set1.\ C1 \ + \ set0.\ C0$$

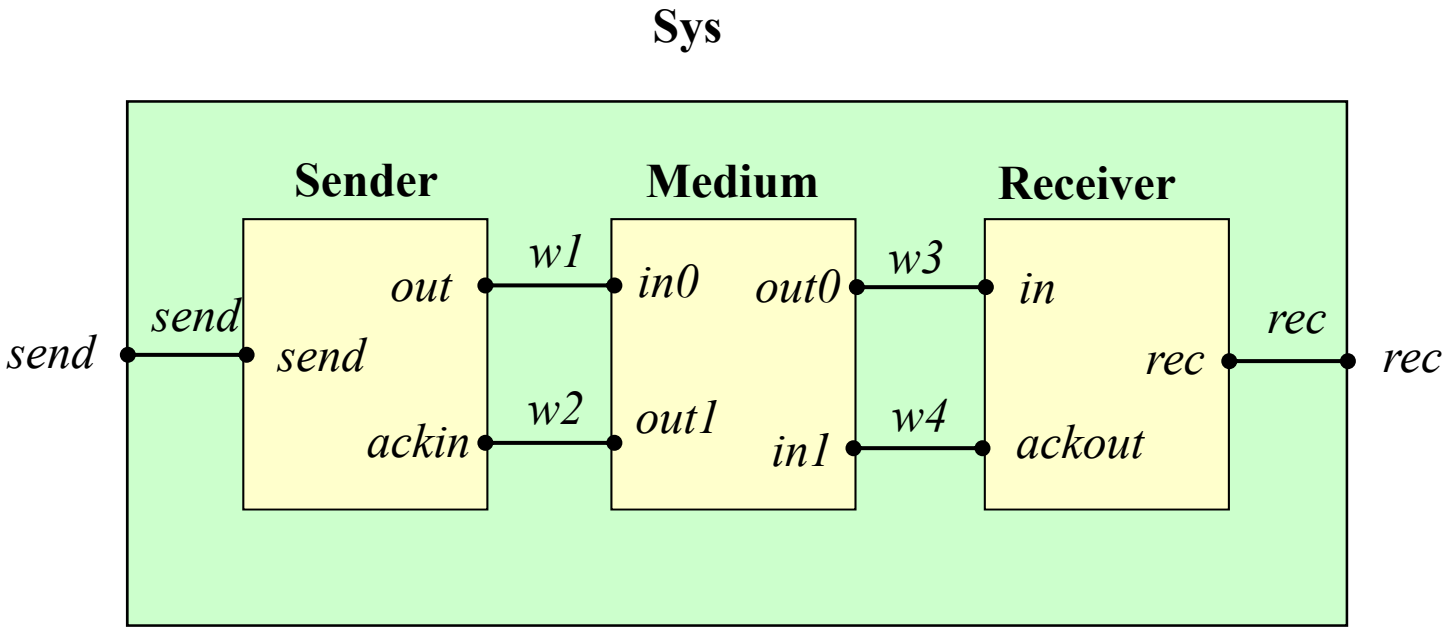
$$C1 = \overline{is1}.\ C1 \ + \ set0.\ C0 \ + \ set1.\ C1$$

Αναπαράσταση δομής συστήματος

- Πως μπορούμε να περιγράψουμε τη δομή ενός συστήματος στη CCS;



Κωδικοποίηση αρχιτεκτονικής



$Sys = (\text{Sender}[w1/out, w2/ackin]$

$| \text{Medium}[w1/in0, w2/out1, w3/out0, w4/in1]$

$| \text{Receiver}[w3/in, w4/ackout]$

$) \setminus \{w1, w2, w3, w4\}$

Επέκταση – CCS με πέρασμα τιμών

- Είσοδος δεδομένου: $a(x).E$

  Διάβασε κάποιο δεδομένο, έστω v, στο κανάλι a και αντικατάστησε στο E όπου x το v.

- Έξοδος δεδομένου: $\overline{a}(e).E$

  Στείλε την τιμή της έκφρασης e στο κανάλι a και προχώρησε ως E.

- Νέοι κανόνες: Έστω Val(e) η τιμή της έκφρασης e.

Αντικατάστησε στο E όπου x το v.

ActIn $$\frac{}{a(x).E \xrightarrow{a(v)} E\{v/x\}}$$

ActOut $$\frac{}{\overline{a}(e).E \xrightarrow{\overline{a}(v)} E}, \quad \mathrm{Val}(e) = v$$

Παράδειγμα

- Καταχωρητής

$$Reg_i \overset{def}{=} \overline{read}(i).Reg_i + write(x).Reg_x$$

$$Reg_{32} \xrightarrow{\overline{read}(32)} Reg_{32}$$

$$Reg_5 \xrightarrow{write(21)} Reg_{21}$$

- Παράλληλη Σύνθεση

$$System \overset{def}{=} Reg_2 \mid Increment \mid Writer_4$$

$$Increment \overset{def}{=} read(i).\overline{write}(i+1).0 \qquad Writer_4 \overset{def}{=} \overline{write}(4).0$$

- Πιθανή Εκτέλεση

$$Reg_2 \mid Increment \mid Write_4 \xrightarrow{\tau(read)} Reg_2 \mid \overline{write}(3).0 \mid Write_4$$

$$\xrightarrow{\tau(write)} Reg_4 \mid \overline{write}(3).0 \mid 0$$

$$\xrightarrow{\tau(write)} Reg_3 \mid 0 \mid 0$$