Αριστοτέλειο
Πανεπιστήμιο
Θεσσαλονίκης

Τεχνητή Νοημοσύνη

Σχεδιασμός Ενεργειών

Ιώαννης Βλαχάβας

Τμήμα Πληροφορικής ΑΠΘ

Θεσσαλονίκη, 2013

Άδειες Χρήσης

Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons. Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα. Το έργο «**Ανοικτά Ακαδημαϊκά Μαθήματα στο Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.

ανοικτάμαθήματα
opencourses

Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σχεδιασμός Ενεργειών (Planning)

Εισαγωγή

- Στο ΜΕΡΟΣ Α παρουσιάστηκε η έννοια του προβλήματος και αντιμετωπίστηκε διεξοδικά η επίλυση προβλημάτων. Επίσης παρουσιάστηκαν (4) τέσσερις κατηγορίες προβλημάτων.
  - Μία κατηγορία προβλημάτων είναι τα **προβλήματα σχεδιασμού ενεργειών** (planning problems) στα οποία η τελική κατάσταση είναι πλήρως γνωστή και επιδιώκεται η εύρεση μίας σειράς ενεργειών, η εκτέλεση των οποίων προκαλεί τη μετάβαση από την αρχική κατάσταση στην τελική.
  - Επίσης, εξετάστηκαν διάφοροι αλγόριθμοι αναζήτησης οι οποίοι όμως κρίνονται ανεπαρκείς για την αντιμετώπιση πραγματικών προβλημάτων αυτού του τύπου.
- Το ΜΕΡΟΣ Γ ασχολείται με ειδικές τεχνικές αναπαράστασης και αλγορίθμους αναζήτησης, για την αποδοτική επίλυση προβλημάτων σχεδιασμού ενεργειών.
  - Αρχικά, παρουσιάζεται η γλώσσα περιγραφής προβλημάτων STRIPS.
  - Στη συνέχεια αναλύονται οι δύο μεγάλες κατηγορίες σχεδιαστών, αυτοί που αναζητούν λύση στο χώρο των καταστάσεων (state-space planners) και αυτοί που αναζητούν λύση στο χώρο των πλάνων (plan-space planners).
  - Ακολουθεί μία παρουσίαση εξελιγμένων τεχνικών σχεδιασμού καθώς και κλασικών σχεδιαστών, όπως ο STRIPS, ο ABSTRIPS, ο DEVISER και ο IPEM.

ΚΕΦΑΛΑΙΟ 15

Βασικές Αρχές και Τεχνικές Σχεδιασμού

Σχεδιασμός Ενεργειών (Planning)

- Προβλήματα σχεδιασμού ενεργειών (planning problems) είναι αυτά στα οποία είναι πλήρως γνωστή η τελική κατάσταση και επιδιώκεται η εύρεση μιας ακολουθίας ενεργειών, μέσω της διαδικασίας του σχεδιασμού ενεργειών (planning).
- Η ακολουθία των ενεργειών που αποτελεί τη λύση ενός προβλήματος σχεδιασμού, ονομάζεται πλάνο (plan) ενώ το πρόγραμμα που την παράγει ονομάζεται σχεδιαστής (planner) ή σύστημα σχεδιασμού (planning system).
- Τα προβλήματα σχεδιασμού διακρίνονται για την υψηλή πολυπλοκότητά τους και τους ιδιαίτερα μεγάλους χώρους αναζήτησης. Αυτό κάνει τους κλασσικούς αλγόριθμους αναζήτησης (που έχουν παρουσιαστεί σε προηγούμενα κεφάλαια), ανεπαρκείς για την αντιμετώπισή τους.
  - Αυτό οφείλεται στο φαινόμενο της συνδυαστικής έκρηξης (τυφλοί αλγόριθμοι) ή στη δυσκολία εφαρμογής κατάλληλων ευριστικών μηχανισμών εξ' αιτίας του τρόπου αναπαράστασης (ευριστικοί αλγόριθμοι).
- Σημαντικό θέμα στην αποδοτικότερη επίλυση προβλημάτων σχεδιασμού, είναι ο ορισμός μίας γλώσσας περιγραφής προβλημάτων, που να υποστηρίζει την εφαρμογή αλγορίθμων ικανών να αντιμετωπίσουν επιτυχώς τα παραπάνω ζητήματα.

Θέματα που θα εξεταστούν

- Αναπαράσταση προβλημάτων- Το μοντέλο STRIPS
- Σχεδιασμός με Αναζήτηση στο Χώρο των Καταστάσεων
- Σχεδιασμός με Αναζήτηση στο Χώρο των Πλάνων
- Εκτέλεση Πλάνων από Πράκτορες Σχεδιαστές

Αναπαράσταση Προβλημάτων

- Ένα πρόβλημα σχεδιασμού ορίζεται από τρεις περιγραφές:
  - Της αρχικής κατάστασης του κόσμου ***Initial***.
  - Των στόχων ***Goals***, που πρέπει να επιτευχθούν.
  - Των διαθέσιμων ενεργειών ***Actions*** που μπορούν να εκτελεστούν, προκειμένου να επιτευχθούν οι στόχοι.
- Η περιγραφή τόσο των καταστάσεων όσο και των ενεργειών καθορίζει αποφασιστικά τα είδη των προβλημάτων σχεδιασμού που μπορεί να περιγραφούν και φυσικά τις δυνατότητες που πρέπει να έχουν τα αντίστοιχα συστήματα σχεδιασμού που θα τα επιλύσουν.
  - θα μπορούσε να χρησιμοποιηθεί η προτασιακή λογική (δεν μπορεί να εκφράσει γενικότητα), ή κατηγορηματική λογική πρώτης τάξης (δεν μπορεί να περιγράψει ενέργειες με μη προκαθορισμένα αποτελέσματα).
- Γενικά υπάρχει ένα μεγάλο φάσμα γλωσσών περιγραφής προβλημάτων σχεδιασμού.
  - Όσο πιο εκφραστική είναι μια αναπαράσταση, τόσο δυσκολότερη είναι η κατασκευή ενός συστήματος σχεδιασμού για την αντιμετώπιση προβλημάτων εκφρασμένων σε αυτή, ενώ παράλληλα αυξάνεται και ο χρόνος που απαιτείται για την επίλυσή τους.

Το Μοντέλο STRIPS
(Stanford Research Institute Planning System)

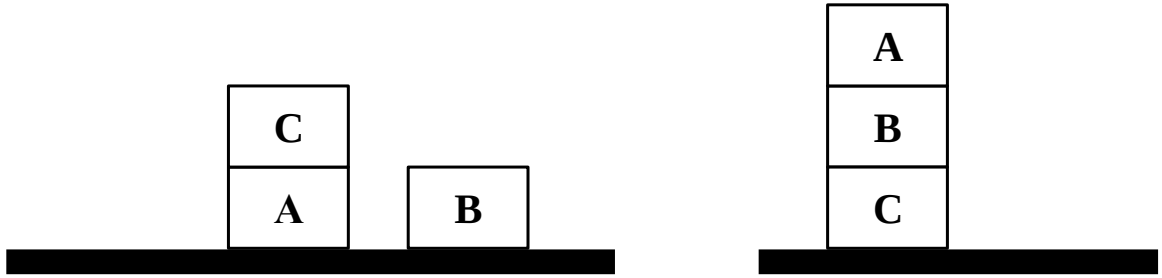
- Το πιο χρησιμοποιημένο μοντέλο περιγραφής προβλημάτων σχεδιασμού.
- Προτάθηκε το 1971, από τους Fikes και Nilsson, ερευνητές στο Stanford, για να καθοδηγεί ένα μικρό ρομπότ (Shakey), στην εκτέλεση διαφόρων απλών ενεργειών.
- Γνώρισε μεγάλη απήχηση, κυρίως λόγω της απλότητας και της φυσικότητας του.
- Έχει στοιχεία προτασιακής λογικής και είναι κατάλληλο για προβλήματα όπου δεν εμφανίζεται αβεβαιότητα.
- Στην αρχική του μορφή δεν υποστήριζε την αναπαράσταση χρονικών και άλλων περιορισμών, ωστόσο στη συνέχεια εμφανίστηκαν πολυάριθμες επεκτάσεις του με πιο πλούσιες εκφραστικές δυνατότητες.

Μοντέλο STRIPS (Παραδοχές)

- Στην απλούστερη μορφή του μοντέλου STRIPS γίνονται οι παρακάτω παραδοχές:
  - *Αδιαίρετες ενέργειες* (*indivisible actions*): Δεν ενδιαφέρει η κατάσταση του κόσμου κατά τη διάρκεια εκτέλεσης μιας ενέργειας, παρά μόνο στην αρχή και στο τέλος αυτής. Επίσης δεν είναι δυνατή η διακοπή της εκτέλεσης μιας ενέργειας ενώ αυτή δεν έχει ολοκληρωθεί.
  - *Προκαθορισμένα αποτελέσματα* (*deterministic effects*): Δεν υπάρχει καμιά αβεβαιότητα όσον αφορά τα αποτελέσματα της εφαρμογής μιας ενέργειας, τα οποία είναι γνωστά εκ των προτέρων.
  - *Πλήρης γνώση* (*omniscience*): Το σύστημα σχεδιασμού έχει πλήρη γνώση για την τρέχουσα κατάσταση του κόσμου αλλά και για τις δικές του δυνατότητες.
  - *Υπόθεση κλειστού συστήματος* (*closed world assumption*): Δεν υπάρχει δυνατότητα προσθήκης νέων ή διαγραφής υπαρχόντων αντικειμένων από τον κόσμο του συστήματος.
  - *Στατικός κόσμος* (*static world*): Ο κόσμος αλλάζει μόνο από τις ενέργειες του συστήματος σχεδιασμού και όχι από μόνος του ούτε από τις ενέργειες κάποιας άλλης οντότητας.

1. Αναπαράσταση Καταστάσεων

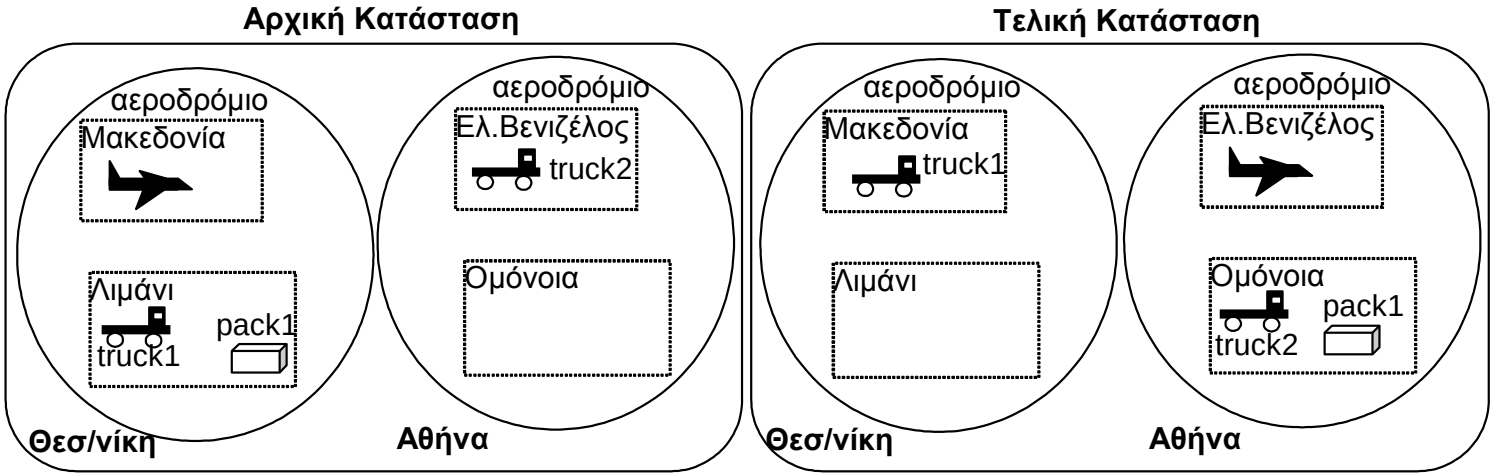
- Στο μοντέλο STRIPS οι καταστάσεις ορίζονται σαν σύνολα από συγκεκριμένα **γεγονότα** (ή προτάσεις) που αληθεύουν.
- Παράδειγμα:



Αρχική κατάσταση Τελική κατάσταση

- Αρχική κατάσταση:
  - Σε φυσική Γλώσσα:
    - Υπάρχει ο κύβος Α.
    - Υπάρχει ο κύβος C.
    - Ο κύβος C βρίσκεται πάνω στον κύβο Α.
    - Ο κύβος C έχει ελεύθερη την επάνω έδρα του.
    - κτλ.
  - Σε STRIPS:
    - **block(a) ∧ block(b) ∧ block(c) ∧ on(a,table) ∧ on(c,a) ∧ on(b,table) ∧ clear(b) ∧ clear(c)**
- Τελική κατάσταση:
  - Σε STRIPS: **on(b,c) ∧ on(a,b)**

Παράδειγμα: Ενα τυπικό πρόβλημα μεταφοράς φορτίων



- Το πρόβλημα μεταφοράς φορτίων αποτελεί χαρακτηριστικό παράδειγμα προβλήματος εφοδιαστικής (logistics).
  - Έστω δύο πόλεις, Θεσσαλονίκη και Αθήνα.
  - Κάθε πόλη έχει δύο τοποθεσίες, η μία είναι το αεροδρόμιό της και η άλλη το κέντρο της.
  - Κάθε πόλη διαθέτει ένα φορτηγό, το οποίο μπορεί να μετακινείται μεταξύ των δύο τοποθεσιών της πόλης, αλλά όχι από τη μία πόλη στην άλλη.
  - Υπάρχει ένα αεροπλάνο, το οποίο μπορεί να μετακινείται μεταξύ των δύο αεροδρομίων.
  - Τέλος, υπάρχει ένα φορτίο, το οποίο αρχικά βρίσκεται στο κέντρο της Θεσσαλονίκης.
  - **Στόχος** είναι η μεταφορά του φορτίου από το κέντρο της Θεσσαλονίκης στο κέντρο των Αθηνών.

Παράδειγμα (συνέχεια): Αναπαράσταση Καταστάσεων

- Η αρχική κατάσταση του παραδείγματος μπορεί να περιγραφεί ως εξής:
  - Υπάρχει η πόλη Θεσσαλονίκη.
  - Υπάρχει η τοποθεσία λιμάνι.
  - Υπάρχει η τοποθεσία Μακεδονία.
  - Η τοποθεσία λιμάνι βρίσκεται στη Θεσσαλονίκη.
  - Η τοποθεσία Μακεδονία είναι αεροδρόμιο.
  - Υπάρχει ένα φορτηγό truck1.
  - κτλ.
- Σε περιγραφή STRIPS:
  - Η αρχική κατάσταση αναπαρίσταται με τα **γεγονότα** (προτάσεις):

    city(thessalonini) $\wedge$ location(harbor) $\wedge$ location(makedonia) $\wedge$
    at_city(harbor,thessaloniki) $\wedge$ airport(makedonia) $\wedge$ truck(truck1) $\wedge$
    at(truck1, harbor) $\wedge$ at(pack1, harbor) $\wedge$ …
  - ενώ η τελική κατάσταση αναπαρίσταται με τις προτάσεις:

    city(thessalonini) $\wedge$ location(harbor) $\wedge$ location(makedonia) $\wedge$
    at_city(harbor,thessaloniki) $\wedge$ airport(makedonia) $\wedge$ truck(truck1) $\wedge$
    at(truck1, makedonia) $\wedge$ at(pack1, omonoia) $\wedge$ …

2. Αναπαράσταση Ενεργειών

- **Ενέργειες (Actions)** είναι δράσεις που επιφέρουν αλλαγή στην κατάσταση του κόσμου. Για την εφαρμογή των ενεργειών απαιτείται η ικανοποίηση ορισμένων συνθηκών-προϋποθέσεων.
- Στην STRIPS αναπαράσταση, μια *ενέργεια* (action) *a* περιγράφεται με τρεις λίστες γεγονότων:

  - Λίστα προϋποθέσεων (Precondition list, Pre(a))
    - Τα γεγονότα που πρέπει να περιλαμβάνονται σε μια κατάσταση, ώστε η ενέργεια να είναι εφαρμόσιμη στην κατάσταση αυτή.

  - Λίστα προσθήκης (Add list, Add(a))
    - Τα γεγονότα που προσθέτει η ενέργεια στη νέα κατάσταση.

  - Λίστα διαγραφής (Delete list, Del(a))
    - Τα γεγονότα της τρέχουσας κατάστασης που δε συμπεριλαμβάνονται στη νέα.
    - Για τη λίστα διαγραφής πρέπει να ισχύει $Del(a) \subseteq Pre(a)$.

Αναπαράσταση Ενεργειών: Παράδειγμα

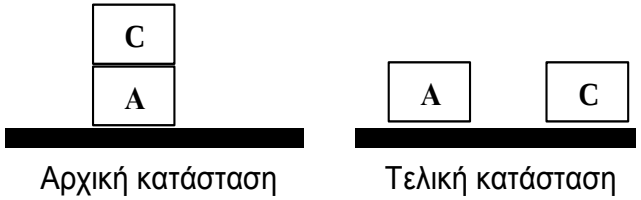
Όνομα ενέργειας	**move_C_from_A_to_table** **μετακίνησε τον κύβο C από τον κύβο A στο τραπέζι**
Λίστα προϋποθέσεων	**block(a), block(c), clear(c), on(c,a)**
Λίστα προσθήκης	**clear(a), on(c,table)**
Λίστα διαγραφής	**on(c,a)**

Αρχική Κατάσταση

block(a)
block(c)
clear(c)
on(c,a)
:
:

Τελική Κατάσταση

block(a)
block(c)
clear(a)
on(c,table)
:
:



Αναπαράσταση Ενεργειών: Παράδειγμα

Όνομα ενέργειας	**move_C_from_A_to_table** **μετακίνησε τον κύβο C από τον κύβο A στο τραπέζι**
Λίστα προϋποθέσεων	**block(a), block(c), clear(c), on(c,a)**
Λίστα προσθήκης	**clear(a), on(c,table)**
Λίστα διαγραφής	**on(c,a)**

Αρχική Κατάσταση

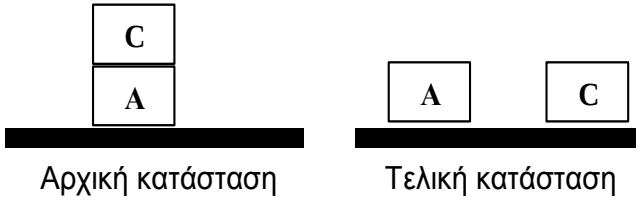
block(a)
block(c)
clear(c)
~~**on(c,a)**~~
:
:

Διαγραφή

Τελική Κατάσταση

block(a)
block(c)
clear(a)
on(c,table)
:
:

Προσθήκη



Σχήματα Ενεργειών

- Επειδή οι ενέργειες ενός προβλήματος σχεδιασμού είναι πάρα πολλές και είναι δύσκολο να απαριθμηθούν, συνηθίζεται να ομαδοποιούνται σε *σχήματα ενεργειών* (*action schemas*) ή *τελεστές* (*operators*).
- Αυτά μπορεί να γίνουν κανονικές ενέργειες με ανάθεση συγκεκριμένης τιμής στις μεταβλητές τους.
- Με τα σχήματα ενεργειών (τελεστών) μειώνεται ο αριθμός των ενεργειών που πρέπει να περιγραφούν.

Παράδειγμα: Ενέργειες στον κόσμο των κύβων

- Στο πρόβλημα των τριών κύβων, ένα σχήμα ενεργειών θα μπορούσε να περιλαμβάνει τις ενέργειες που μετακινούν έναν κύβο από την κορυφή μιας στοίβας στο τραπέζι.
  - Υπάρχουν έξι τέτοιες ενέργειες, ανάλογα με το ποιος είναι ο κύβος που μετακινείται και ποιος κύβος βρίσκεται από κάτω του, οι οποίες μπορούν να παρασταθούν από το ακόλουθο σχήμα ενεργειών:

Όνομα (σχήματος) ενέργειας (ή τελεστή)	**move_X_from_Y_to_table** **μετακίνησε έναν κύβο X από τον κύβο Y στο τραπέζι**
Λίστα προϋποθέσεων	**block(X), block(Y), clear(X), on(X,Y)**
Λίστα προσθήκης	**clear(Y), on(X,table)**
Λίστα διαγραφής	**on(X,Y)**

Παράδειγμα: Ενέργειες στη μεταφορά φορτίων

- Στο πρόβλημα των φορτίων οι επιτρεπτές ενέργειες είναι:
  - *Φόρτωσε το φορτίο στο φορτηγό.*
  - *Ξεφόρτωσε το φορτίο από το φορτηγό.*
  - *Φόρτωσε το φορτίο στο αεροπλάνο.*
  - *Ξεφόρτωσε το φορτίο από το αεροπλάνο.*
  - *Μετακίνησε το φορτηγό.*
  - *Μετακίνησε το αεροπλάνο.*
- Τα παραπάνω μπορούν να περιγραφούν με σχήματα ενεργειών, τα οποία συγκεκριμενοποιούνται σε ενέργειες ανάλογα με τις τιμές που παίρνουν οι μεταβλητές τους.
- Για παράδειγμα, το σχήμα ενέργειας "*φόρτωσε το φορτίο στο φορτηγό*" μπορεί να οριστεί ως εξής:

Όνομα σχήματος ενέργειας (τελεστή)	*load_truck*(*T*,*P*,*L*) (φόρτωσε το φορτίο *P* στο φορτηγό *T* στην τοποθεσία *L*)
Λίστα προϋποθέσεων	*package*(*P*), *truck*(*T*), *location*(*L*), *at*(*T*,*L*), *at*(*P*,*L*)
Λίστα προσθηκών	*in*(*P*,*T*)
Λίστα διαγραφών	*at*(*P*,*L*)

Πρόβλημα σχεδιασμού (Planning problem): Ορισμός (1/2)

- Με βάση τα παραπάνω, ένα πρόβλημα σχεδιασμού *P* μπορεί να παρασταθεί με μια τριπλέτα *P=(Actions, Initial, Goals)*, όπου *Actions* το σύνολο των συγκεκριμένων ενεργειών, *Initial* η *αρχική κατάσταση* και *Goals* οι *στόχοι*.
  - Οι στόχοι (*Goals*) του προβλήματος ορίζονται σα σύζευξη προτάσεων (γεγονότων) που πρέπει να αληθεύουν και αποτελούν υποσύνολο της τελικής κατάστασης.
  - Το πρόβλημα συνίσταται στην εύρεση μιας ακολουθίας ενεργειών a_1, a_2, ..., a_N η οποία να είναι εφαρμόσιμη στην αρχική κατάσταση και η κατάσταση που προκύπτει μετά την εφαρμογή της να είναι υπερσύνολο των στόχων.
- Για να είναι εφαρμόσιμη μια ενέργεια *a* σε μια κατάσταση *S* πρέπει να ισχύει:
  - $Pre(a) \subseteq S$
  - Δηλαδή τα γεγονότα που αναφέρονται στη λίστα προϋποθέσεων της ενέργειας a να είναι υποσύνολο των γεγονότων που περιγράφουν την κατάσταση S.
- Η κατάσταση *S'* που προκύπτει μετά την εφαρμογή της ενέργειας *a* στην κατάσταση *S* ορίζεται ως:
  - $S' = result(S, a) = S - Del(a) \cup Add(a)$
  - Δηλαδή η νέα κατάσταση *S'* προκύπτει με την αφαίρεση από την κατάσταση *S* των γεγονότων της λίστας διαγραφών και την προσθήκη των γεγονότων της λίστας προσθηκών της ενέργειας *a*.

Πρόβλημα σχεδιασμού (Planning problem): Ορισμός (2/2)

- Επαγωγικά μπορεί να οριστεί η κατάσταση που προκύπτει μετά την εφαρμογή μιας ακολουθίας ενεργειών $\langle a_1, a_2, ..., a_N \rangle$ σε μια κατάσταση S ως εξής:
  - $S' = result(S, \langle a_1, a_2, ..., a_N \rangle) = result(result(S, \langle a_1, a_2, ..., a_{N-1} \rangle), a_N)$
  - Προϋπόθεση: Κάθε ενέργεια a_i είναι εφαρμόσιμη στην κατάσταση $result(S, \langle a_1, a_2, ..., a_{i-1} \rangle)$, για κάθε i=1, 2, ..., N

- Οι ακολουθίες ενεργειών ονομάζονται ***πλάνα (plans)***.
  - Ένα πλάνο το οποίο μπορεί να εφαρμοστεί στην αρχική κατάσταση ονομάζεται ***έγκυρο πλάνο (valid plan)***.
  - Ένα έγκυρο πλάνο το οποίο πετυχαίνει τους στόχους ονομάζεται ***λύση (solution)*** του προβλήματος σχεδιασμού.

- Ένα πρόβλημα σχεδιασμού μπορεί να έχει μία ή περισσότερες ή καμία λύση.
  - Στην τελευταία περίπτωση το πρόβλημα σχεδιασμού χαρακτηρίζεται ως *άλυτο* (*unsolvable*

Διαγραμματική Αναπαράσταση πλάνων (1/3)

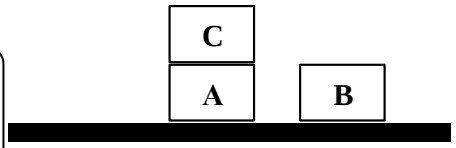
- Διαγραμματική αναπαράσταση **ενέργειας**:

move_C_from_A_to_table

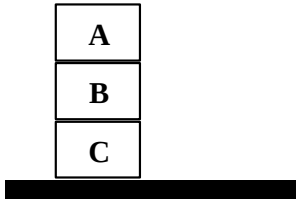
`block(c)`	`- on(c,a)`
`block(a)`	`+ on(c,table)`
`on(c,a)`	
`clear(c)`	

Προϋποθέσεις

Αποτελέσματα
- Del
+ Add



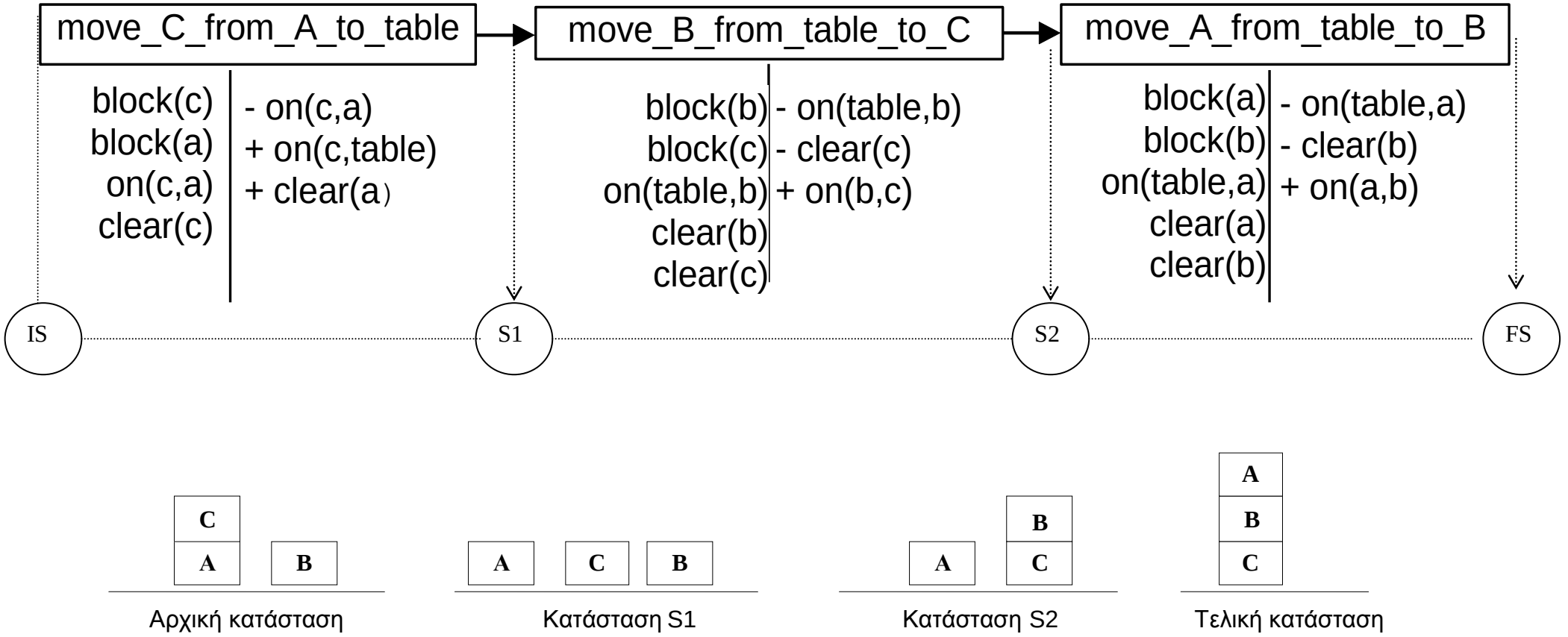
Αρχική κατάσταση



Τελική κατάσταση

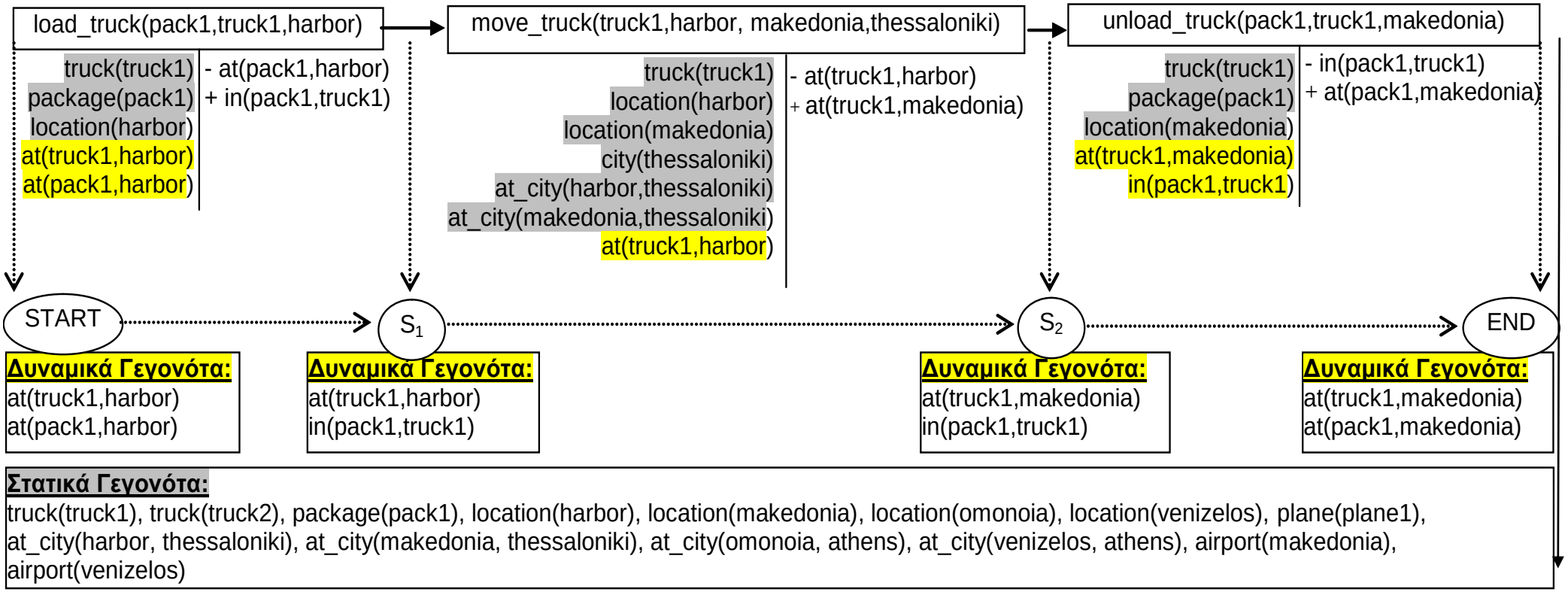
Διαγραμματική Αναπαράσταση πλάνων (1/2)

- Διαγραμματική αναπαράσταση **πλάνου** με δίκτυο ενεργειών (*procedural network*) όπου οι κόμβοι του είναι οι ενέργειες, ενώ οι (κατευθυνόμενες) ακμές δηλώνουν την ακολουθία των ενεργειών (Λύση του προβλήματος των 3 κύβων).



Παράδειγμα: Πλάνα στη μεταφορά φορτίων

- Πλάνο τριών ενεργειών, το οποίο μετακινεί το φορτίο *pack1* από τη θέση *harbor* της Θεσσαλονίκης στη θέση αεροδρόμιο *makedonia* της Θεσσαλονίκης, με τη χρήση του φορτηγού *truck1*.
  - Στο σχήμα φαίνονται τα **δυναμικά** γεγονότα σε κάθε κατάσταση, δηλαδή αυτά που αλλάζουν κατά την εκτέλεση του πλάνου, και τα **στατικά** γεγονότα, δηλαδή αυτά που παραμένουν αμετάβλητα από την αρχική έως την τελική κατάσταση.



Διαγραμματική Αναπαράσταση πλάνων (2/2)

- Διαγραμματική αναπαράσταση πλάνου με ραβδόγραμμα (Gantt bar chart) όπου κάθε ράβδος είναι μία ενέργεια και το μήκος της ράβδου η διάρκειά της.

- *Γραμμικό πλάνο* (*linear plan*): Υπάρχει αυστηρή διαδοχή των ενεργειών.

- *Μη-γραμμικό πλάνο* (*non-linear plan*): Δεν υπάρχει αυστηρή διαδοχή ενεργειών, αλλά υπάρχει η δυνατότητα δύο ενέργειες να εκτελούνται παράλληλα, με μερική ή ολική χρονική αλληλοεπικάλυψη.

Αναπαράσταση STRIPS: Μειονεκτήματα

- Δεν αναφέρει το χρόνο κατά τον οποίο ισχύουν τα γεγονότα.
  - Θα έπρεπε να προστεθεί ένας χρονικός προσδιορισμός σε κάθε ένα από τα γεγονότα της κατάστασης.
- Δεν μπορεί να περιγράψει συνεχείς μεταβολές.
- Θεωρεί πλήρη βεβαιότητα για την ισχύ των γεγονότων.
  - Για παράδειγμα, το γεγονός "Ο κύβος C βρίσκεται πάνω στον κύβο Α" θα μπορούσε να ισχύει με βεβαιότητα 80%.
  - Η δήλωση της βεβαιότητας των γεγονότων μπορεί να γίνει με συντελεστές.
- Δεν είναι πλήρης.
  - Δεν περιέχει γνώση για όλες τις παραμέτρους του προβλήματος αλλά μόνο για αυτές που θεωρείται ότι μπορούν να επηρεάσουν τη λύση του.
  - Για παράδειγμα, στην περιγραφή της αρχικής κατάστασης δεν δηλώνονται τα χρώματα των κύβων ούτε η εξωτερική θερμοκρασία.
  - Τα γεγονότα που συμπεριλαμβάνονται στην αναπαράσταση ενός προβλήματος αποτελούν το *πλαίσιό* του (*frame*).

Σχεδίαση με λογισμό καταστάσεων (situation calculus)

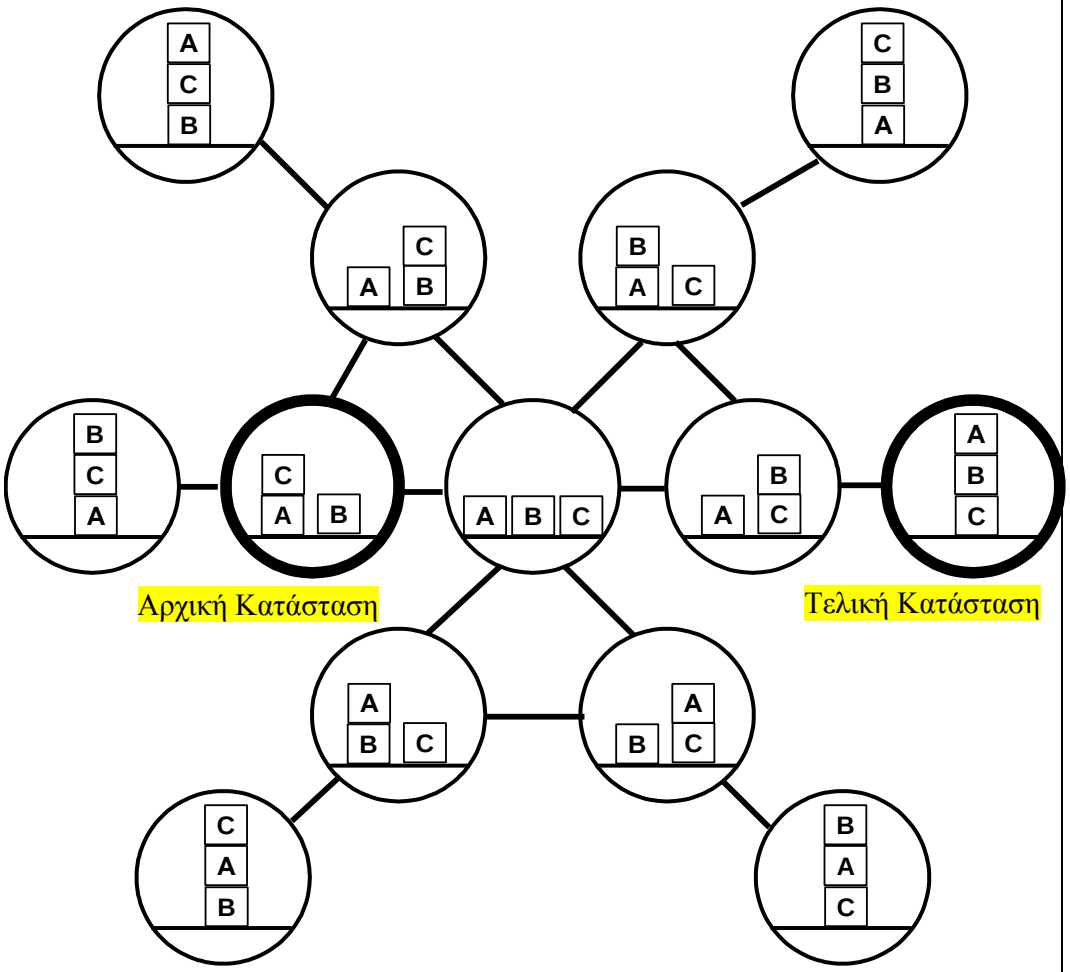
- Επικρατούσε πριν από το σχεδιαστή STRIPS.
- Κάθε ενέργεια έπρεπε να ορίζει με σαφήνεια ολόκληρη την κατάσταση που θα έχει ο κόσμος μετά την εκτέλεσή της.
  - Για παράδειγμα, μια ενέργεια που μετακινεί το φορτηγό truck1, έπρεπε να καθορίζει όχι μόνο τη νέα θέση του μετακινούμενου φορτηγού, αλλά επίσης και το γεγονός ότι το φορτηγό truck2 ή το αεροπλάνο plane1 έμειναν στη θέση τους.
- Για κάθε ενέργεια γράφονταν πάρα πολλά αξιώματα, τα λεγόμενα *αξιώματα του πλαισίου* (frame axioms), τα οποία καθόριζαν ποιες από τις προτάσεις του πλαισίου του προβλήματος παρέμεναν ανεπηρέαστες κατά την εκτέλεση της ενέργειας.
  - Το πρόβλημα με τα πολλά αξιώματα πλαισίου έμεινε γνωστό σαν το *πρόβλημα του πλαισίου* (frame problem).
- Η αναπαράσταση των ενεργειών στον STRIPS έλυσε σε μεγάλο βαθμό το πρόβλημα του πλαισίου, δηλώνοντάς για κάθε ενέργεια μόνο τις προτάσεις που αυτή αλλάζει και θεωρώντας σιωπηρά ότι όλες οι υπόλοιπες προτάσεις παραμένουν ανεπηρέαστες.

Σχεδιασμός με Αναζήτηση στο Χώρο των Καταστάσεων

- Ο πιο απλός τρόπος αντιμετώπισης ενός προβλήματος σχεδιασμού είναι με αναζήτηση στο χώρο των καταστάσεων, χρησιμοποιώντας κάποιον από τους γνωστούς αλγορίθμους αναζήτησης (Πλεονέκτημα).
- Οι αντίστοιχοι σχεδιαστές ονομάζονται *σχεδιαστές χώρου καταστάσεων* (*state-space planners*), γιατί σε κάθε επανάληψη του αλγορίθμου αναζήτησης επιλέγουν μια κατάσταση του χώρου καταστάσεων, από την οποία παράγουν νέες καταστάσεις με την εφαρμογή κάποιων ενεργειών.

Σχεδιασμός με Αναζήτηση στο Χώρο των Καταστάσεων

- Στο σχήμα φαίνονται όλες οι καταστάσεις του προβλήματος των τριών κύβων.
  - Κάθε κόμβος του γράφου παριστάνει μια κατάσταση, ενώ ακμές μεταξύ τους δηλώνουν ότι η μια κατάσταση μπορεί να επιτευχθεί με εφαρμογή κάποιας ενέργειας στην άλλη.
  - Γενικά οι ακμές είναι κατευθυνόμενες, ωστόσο εδώ σχεδιάστηκαν χωρίς βέλη, υπονοώντας διπλή κατεύθυνση, επειδή στο συγκεκριμένο πρόβλημα όλες οι ενέργειες είναι αντιστρέψιμες.



Κατεύθυνση Διάσχισης του Χώρου των Καταστάσεων

- Κατά την παρουσίαση των αλγορίθμων αναζήτησης θεωρήθηκε ότι η αναζήτηση στο χώρο των καταστάσεων πραγματοποιείται ξεκινώντας από την αρχική κατάσταση και προχωρώντας προς μια τελική.
  - Αυτή η κατεύθυνση διάσχισης του χώρου των καταστάσεων ονομάζεται ορθή διάσχιση (progression).
  - Υπάρχει και η δυνατότητα διάσχισης του χώρου των καταστάσεων από την τελική κατάσταση προς την αρχική η οποία ονομάζεται ανάστροφη διάσχιση (regression).

Είδη Διάσχισης

- Ορθή διάσχιση (progression)
  - Η αναζήτηση στο χώρο των καταστάσεων πραγματοποιείται ξεκινώντας από την αρχική κατάσταση και προχωρώντας προς μια τελική.
- Ανάστροφη διάσχιση (regression)
  - Η διάσχιση γίνεται από τους στόχους προς την αρχική κατάσταση.
  - Αναφέρεται και ως: *Τεχνική ανάλυσης των μέσων και των στόχων* (means-ends analysis): Επικεντρώνεται στην εύρεση εκείνων των ενεργειών (μέσων) που επιτυγχάνουν τους στόχους.
- Διάσχιση Διπλής Κατεύθυνσης (bi-directional)

Ορθή Διάσχιση

- Αρχίζοντας από την αρχική κατάσταση εφαρμόζονται όλες οι ενέργειες που μπορούν να εφαρμοστούν και δημιουργούν νέες καταστάσεις.
- Από τις καταστάσεις αυτές επιλέγεται μία και επαναλαμβάνεται η ίδια διαδικασία έως ότου προκύψει η τελική κατάσταση.

- Έστω μια αρχική κατάσταση (I), ένα σύνολο στόχων (G) και ένα σύνολο ενεργειών.

- Επιλέγεται μια ενέργεια a της οποίας οι προϋποθέσεις της εμπεριέχονται (είναι υποσύνολο) στην αρχική κατάσταση (I).

- Ύστερα από την εφαρμογή της ενέργειας, προκύπτει μια νέα κατάσταση S:
  - $S = I - Del(a) \cup Add(a)$

- Η διαδικασία εφαρμόζεται επαναληπτικά στη νέα κατάσταση S, μέχρις ότου βρεθεί μια κατάσταση που είναι υπερσύνολο (εμπεριέχει) των στόχων.

Ανάστροφη Διάσχιση (1/2)

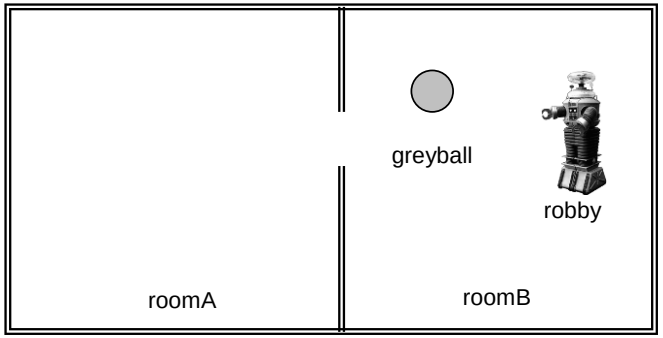
- Στην περίπτωση αυτή, η διάσχιση γίνεται από τους στόχους προς την αρχική κατάσταση.
  - Διαισθητικά μπορεί κανείς να αντιληφθεί το σχεδιασμό με ανάστροφη διάσχιση με το εξής παράδειγμα:
  - Εάν ο απώτερος στόχος κάποιου ανθρώπου είναι να φάει, πρέπει πρώτα να μαγειρέψει, άρα πρέπει νωρίτερα να ψωνίσει, άρα πρέπει πιο πριν να πάει στο μπακάλικο, κτλ.
  - Η χρήση της ανάστροφης διάσχισης υπήρξε δημοφιλής από τα πρώτα συστήματα σχεδιασμού, αφού υιοθετήθηκε από τα συστήματα GPS και STRIPS.
  - Αναφέρεται μάλιστα και ως τεχνική ανάλυσης των μέσων και των στόχων (means-ends analysis), αφού επικεντρώνεται στην εύρεση εκείνων των ενεργειών (μέσων) που επιτυγχάνουν τους στόχους, πράγμα το οποίο δεν γίνεται εύκολα στην ορθή διάσχιση.

Ανάστροφη Διάσχιση (2/2): **Εκτέλεση**

- Έστω μια αρχική κατάσταση (I), ένα σύνολο στόχων (G) και ένα σύνολο ενεργειών.
- Επιλέγεται μια ενέργεια, έστω *a*, τέτοια ώστε κανένα από τα γεγονότα που αυτή διαγράφει να μην εμφανίζεται στην τελική κατάσταση (σύνολο στόχων), ενώ πρέπει (στην τελική κατάσταση) να εμφανίζεται τουλάχιστον ένα από τα γεγονότα που αυτή προσθέτει.
  - $Del(a) \cap G = \emptyset$ **και** $Add(a) \cap G \neq \emptyset$
- Ύστερα από την εφαρμογή της ενέργειας, το σύνολο των στόχων αναθεωρείται σε ένα νέο σύνολο στόχων G', το οποίο ισούται με:
  - $G' = Pre(a) \cup G - Add(a)$
- Η διαδικασία εφαρμόζεται επαναληπτικά στο νέο σύνολο στόχων (G'), μέχρις ότου βρεθεί ένα σύνολο γεγονότων που (να περιέχεται στην) να είναι υποσύνολο της αρχικής κατάστασης.
- Τα σημεία επιλογής των ενεργειών είναι σημεία οπισθοδρόμησης backtracking points), στα οποία η αναζήτηση μπορεί να επιστρέψει για την επιλογή κάποιας άλλης ενέργειας, εφόσον οι προηγούμενες επιλογές οδήγησαν σε αδιέξοδο.

Παράδειγμα: Ένα πρόβλημα κίνησης και λαβής από ρομπότ

- Έστω το πρόβλημα του παρακάτω σχήματος στο οποίο υπάρχουν δύο δωμάτια που συνδέονται μεταξύ τους, μία μπάλα η οποία πρέπει να μεταφερθεί από το ένα δωμάτιο στο διπλανό και ένα ρομπότ που μπορεί να κινείται μέσα στο χώρο και να μεταφέρει αντικείμενα.



- Η αρχική κατάσταση START του προβλήματος περιγράφεται από τα γεγονότα:

START={robot(robby) ∧ room(roomA) ∧ room(roomB) ∧ ball(greyball) ∧ at(robby,roomB) ∧ at(greyball,roomB) ∧ free(robby)}

- ενώ επειδή υπάρχει μόνο ένας στόχος, η κατάσταση END που είναι:

END = {at(greyball,roomA)}.

- Οι τελεστές (σχήματα ενεργειών) είναι:

A1

move (R, X, Y)	
robot(R) room(X) room(Y) at(R,X)	- at(R,X) + at(R,Y)

A2

pick_ball (R,B,X)	
robot(R) ball(B) room(X) at(R,X) at(B,X) free(R)	- at(B,X) - free(R) + has(R,B)

A3

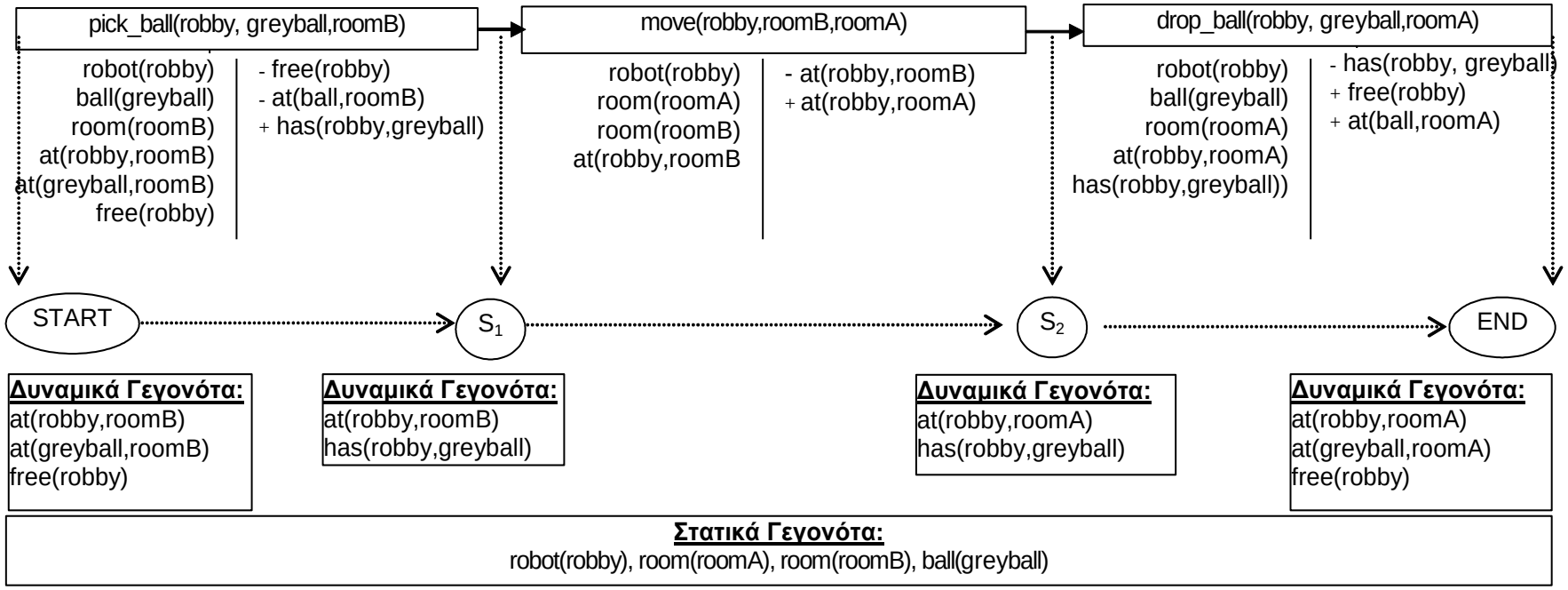
drop_ball (R,B,X)	
robot(R) ball(B) room(X) at(R,X) has(R,B)	- has(R,B) + at(B,X) + free(R)

Επίλυση με ορθή διάσχιση (1/2)

- Η αναζήτηση ξεκινάει από την αρχική κατάσταση και το πρώτο βήμα είναι η εύρεση των ενεργειών που μπορεί να εφαρμοστούν σε αυτήν.
- **Βήμα 1**: Καθώς $prec(A_1) \subseteq START$ και $prec(A_2) \subseteq START$, μπορεί να εφαρμοστούν και οι δύο ενέργειες:
  - $A_1 = move(robby, roomB, roomA)$
  - $A_2 = pick\_ball(robby, greyball, roomB)$
  - Έστω ότι ο αλγόριθμος αναζήτησης επιλέγει την ενέργεια A_1, οπότε η νέα κατάσταση S_A που προκύπτει είναι η:
    - $S_A = START - del(A_1) \cup add(A_1)$
      $= \{ robot(robby) \wedge room(roomA) \wedge room(roomB) \wedge ball(greyball) \wedge$
      $at(robby, roomA) \wedge at(greyball, roomB) \wedge free(robby) \}$
- **Βήμα 2**: Στην κατάσταση S_A μπορεί να εφαρμοστεί μόνο η ενέργεια
  - $A_3 = move(robby, roomA, roomB)$
  - από την οποία προκύπτει η κατάσταση
    - $S_B = \{ robot(robby) \wedge room(roomA) \wedge room(roomB) \wedge ball(greyball) \wedge$
      $at(robby, roomB) \wedge at(greyball, roomB) \wedge free(robby) \}$

Επίλυση με ορθή διάσχιση (2/2)

- **Βήμα 3**: Επειδή η κατάσταση S_B έχει ήδη εξεταστεί ($S_B \equiv START$), ο αλγόριθμος επιλέγει την ενέργεια A_2 και προκύπτει η κατάσταση:
  - $S_C = START - del(A_2) \cup add(A_2)$
    $= \{ robot(robby) \wedge room(roomA) \wedge room(roomB) \wedge ball(greyball) \wedge$
    $at(robby, roomA) \wedge has(robby, greyball) \}$
  - Η ίδια διαδικασία επαναλαμβάνεται μέχρι να βρεθεί κατάσταση S_F για την οποία να ισχύει $END \subseteq S_F$.
- Ένα πλάνο που θα μπορούσε να βρεθεί για το συγκεκριμένο πρόβλημα είναι:



Επίλυση με ανάστροφη διάσχιση (1/2)

- Η αναζήτηση ξεκινάει από τους στόχους και το πρώτο βήμα είναι η εύρεση των ενεργειών που μπορεί να εφαρμοστούν σε αυτούς.
- **Βήμα 1**: Η μοναδική ενέργεια που προσθέτει το στόχο *at*(*greyball*,*RoomA*) είναι η:
  - A_1 = *drop_ball*(*robby*,*greyball*,*roomA*)
  - για την οποία αποδεικνύεται επίσης ότι $Del(A_1) \cap END = \varnothing$. Επομένως, ο αλγόριθμος την εφαρμόζει ανάστροφα στο *END* και προκύπτει το νέο σύνολο στόχων G_A:
    - $G_A = Pre(A_1) \cup END - Add(A_1)$ = { *at*(*robby*, *roomA*), *has*(*robby*, *greyball*) }
- **Βήμα 2**: Στο σύνολο G_A μπορεί να εφαρμοστούν ανάστροφα οι ενέργειες:
  - A_2 = *move*(*robby*, *roomB*, *roomA*)
  - A_3 = *pick_ball*(*robby*, *greyball*, *roomB*)
  - A_4 = *pick_ball*(*robby*, *greyball*, *roomA*).
  - Έστω ότι επιλέγεται η <u>ενέργεια A_3,</u> οπότε το νέο σύνολο στόχων που προκύπτει είναι:
    - G_B = {*at*(*robby*,*roomA*), *at*(*robby*,*roomB*), *at*(*greyball*,*roomB*), *free*(*robby*)}
  - Εύκολα όμως προκύπτει ότι το G_B δεν είναι έγκυρο, καθώς τα γεγονότα:
    - *at*(*robby*, *roomA*) και *at*(*robby*, *roomB*)
  - είναι ασύμβατα μεταξύ τους. Οπότε ο αλγόριθμος οπισθοδρομεί στο προηγούμενο βήμα.

Επίλυση με ανάστροφη διάσχιση (2/2)

- **Βήμα 3**: Έστω ότι ο αλγόριθμος επιλέγει τώρα την ενέργεια A_2, οπότε το νέο σύνολο στόχων που προκύπτει είναι το:
  - G_C = { *at*(*Rrobby*, *roomB*), *has*(*robby*, *greyball*) }
- Η ίδια διαδικασία επαναλαμβάνεται μέχρι να βρεθεί σύνολο στόχων, το οποίο να περιέχει όλα τα δυναμικά γεγονότα της αρχικής κατάστασης.
- Το πλάνο που τελικά θα προκύψει από την ανάστροφη διάσχιση, ταυτίζεται με αυτό της ορθής.

Σύγκριση Ορθής και Ανάστροφης Διάσχισης (1/2)

- Από τη σύγκριση των δύο κατευθύνσεων αναζήτησης προκύπτει το συμπέρασμα ότι οι δύο κατευθύνσεις εμφανίζουν την ίδια τάξη πολυπλοκότητας.
  - Με την εφαρμογή ενός αλγορίθμου ευριστικής αναζήτησης ή του αλγορίθμου πρώτα σε βάθος, και με απόλυτη επιτυχία στην επιλογή των σωστών ενεργειών και οι δύο κατευθύνσεις θα εκτελέσουν τον ίδιο αριθμό επαναλήψεων πριν βρουν το πλάνο.
  - Ωστόσο, σε μια πραγματική υλοποίηση δεν υπάρχει απόλυτη επιτυχία στην επιλογή των σωστών ενεργειών, οπότε αυτό το οποίο επηρεάζει σημαντικά τον όγκο της αναζήτησης είναι ο αριθμός των εφαρμόσιμων ενεργειών σε κάθε κατάσταση του χώρου καταστάσεων, οι οποίες και πρέπει να ελεγχθούν κατά την αναζήτηση.
  - Ο αριθμός αυτός ονομάζεται παράγοντας διακλάδωσης (branching factor) και δεν έχει συνήθως σταθερή τιμή.
  - Αν υποτεθεί ότι η μέση τιμή του είναι b, τότε η πολυπλοκότητα του προβλήματος της αναζήτησης για μεγάλους χώρους είναι της τάξης του O(bn), όπου n το μήκος της λύσης του προβλήματος.
  - Ακόμη και μικρές διαφορές στον παράγοντα διακλάδωσης οδηγούν σε μεγάλες διαφοροποιήσεις στην απόδοση της αναζήτησης.

Σύγκριση Ορθής και Ανάστροφης Διάσχισης (2/2)

- Κάνοντας την εύλογη υπόθεση ότι συνήθως οι στόχοι ενός προβλήματος σχεδιασμού αποτελούν ένα μικρό σύνολο γεγονότων, σε σχέση με αυτά που απαιτούνται για να περιγράψουν πλήρως μια κατάσταση, τότε είναι πολύ πιθανό ο παράγοντας διακλάδωσης στην αναζήτηση με ανάστροφη διάσχιση να είναι μικρότερος από τον αντίστοιχο της αναζήτησης με ορθή διάσχιση, άρα η ανάστροφη διάσχιση είναι συνήθως αποτελεσματικότερη.
  - Αυτό εξηγείται από το ότι υπάρχουν συνήθως πολλές ενέργειες που μπορεί να εφαρμοστούν στην αρχική κατάσταση, ενώ είναι λίγες οι ενέργειες που επιτυγχάνουν τους στόχους.
  - Ωστόσο σε ορισμένες περιπτώσεις εμφανίζεται το αντίθετο φαινόμενο, δηλαδή να υπάρχουν λίγες ενέργειες που μπορεί να εφαρμοστούν στην αρχική κατάσταση και πολλές ενέργειες που επιτυγχάνουν τους στόχους, οπότε και πλεονεκτεί η ορθή διάσχιση.
- Να σημειωθεί τέλος ότι υπάρχει και η τεχνική της αναζήτησης διπλής κατεύθυνσης (bidirectional search), και η οποία προσπαθεί να συνδυάσει τα πλεονεκτήματα των δύο κατευθύνσεων, όπως το σύστημα σχεδιασμού BP.

Σχεδιασμός με Αναζήτηση στο Χώρο των Πλάνων (1/2)

- Στο σχεδιασμό στο χώρο των καταστάσεων θεωρήθηκε ότι οι ενέργειες των πλάνων είναι πλήρως διατεταγμένες, τα δε πλάνα παράγονται προσθέτοντας νέες ενέργειες είτε στο τέλος τους (ορθή διάσχιση) ή στην αρχή τους (ανάστροφη διάσχιση).
  - Τα πλάνα αυτά χαρακτηρίζονται ως γραμμικά πλάνα (linear plans) ή πλάνα πλήρους διάταξης (totally ordered plans).
- Μια εναλλακτική προσέγγιση είναι η αναζήτηση στο *χώρο των πλάνων* (*plan space*). Αυτή η προσέγγιση χρησιμοποιεί τους γνωστούς αλγορίθμους αναζήτησης, ωστόσο:
  - Το μέτωπο αναζήτησης και το κλειστό σύνολο δεν περιέχουν καταστάσεις, αλλά ημιτελή πλάνα, τα οποία δεν αντιστοιχούν σε συγκεκριμένες καταστάσεις.
  - Τα ημιτελή πλάνα είναι σύνολα από ενέργειες, όχι απαραίτητα συγκεκριμένες και όχι απαραίτητα πλήρως διατεταγμένες στο χρόνο.
  - Λύση αποτελεί η εύρεση ενός πλάνου του οποίου οι ενέργειες είναι συγκεκριμένες και για τις οποίες υπάρχει μια τουλάχιστον έγκυρη και πλήρης διάταξή τους στο χρόνο.
  - Η αναζήτηση στο χώρο των πλάνων ξεκινά από ένα κενό πλάνο ενώ η λύση που επιστρέφεται είναι το πλάνο του τελικού κόμβου.

Σχεδιασμός με Αναζήτηση στο Χώρο των Πλάνων (2/2)

- Βασίζεται στη λογική: *Αρχή της ελάχιστης δέσμευσης* (*least commitment principle*): Οι ενέργειες δεν τοποθετούνται σε συγκεκριμένες θέσεις στο χρόνο και οι μεταβλητές τους δε δεσμεύονται σε συγκεκριμένες τιμές αντικειμένων, εφόσον δε συντρέχει λόγος για κάτι τέτοιο.
- Οι σχεδιαστές που αναζητούν λύσεις στο χώρο των πλάνων αποκαλούνται με διάφορες ονομασίες, όπως:
  - σχεδιαστές χώρου πλάνων (plan-space planners),
  - μη-γραμμικοί σχεδιαστές (non-linear planners),
  - σχεδιαστές μερικής διάταξης (partial order planners) αλλά και
  - σχεδιαστές ελάχιστης δέσμευσης (least-commitment planners).
- Ωστόσο, οι όροι "μη-γραμμικός σχεδιαστής" και "σχεδιαστής μερικής διάταξης" αναφέρονται κυρίως στους σχεδιαστές που διατηρούν μερική διάταξη συγκεκριμένων όμως ενεργειών, ενώ οι υπόλοιποι όροι καλύπτουν και τους σχεδιαστές που χειρίζονται μη-συγκεκριμένες ενέργειες.
- Ο πρώτος σχεδιαστής αυτής της κατηγορίας ήταν ο NOAH (1974), ενώ ακολούθησαν πολλοί άλλοι, από τους οποίους ξεχώρισαν οι SNLP (1991) και UCPOP (1992).

Αναπαράσταση Μη-Γραμμικών Πλάνων

- Ένα μη-γραμμικό πλάνο με συγκεκριμένες ενέργειες ορίζεται ως μια τριάδα, (A,O,L), όπου:
  - A είναι ένα σύνολο ενεργειών
  - O είναι ένα σύνολο περιορισμών διάταξης (ordering constraints)
  - L ένα σύνολο αιτιολογικών συνδέσεων (causal links)
- Για παράδειγμα, εάν $A=\{a_1, a_2, a_3\}$, τότε ένα πιθανό σύνολο περιορισμών διάταξης θα ήταν το $O=\{a_1<a_3, a_2<a_3\}$.
  - Το παραπάνω σύνολο περιορισμών διάταξης είναι συμβατό με τις πλήρεις διατάξεις $a_1<a_2<a_3$ αλλά και $a_2<a_1<a_3$.

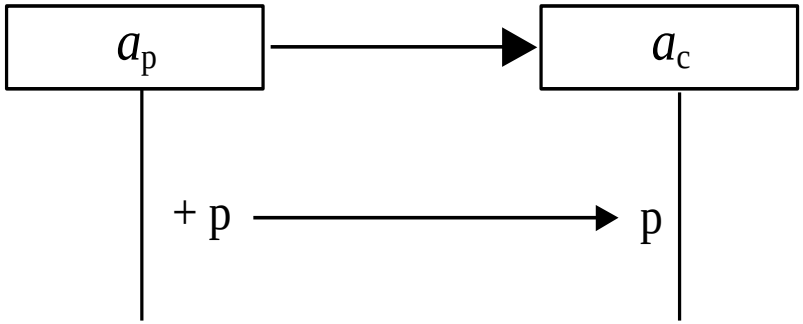
Αιτιολογικές Συνδέσεις (1/2)

- Ένας μη-γραμμικός σχεδιαστής πρέπει να διατηρεί πληροφορίες σχετικά με τους λόγους για τους οποίους μια ενέργεια εισήλθε στο πλάνο, και μάλιστα σε μια συγκεκριμένη θέση.
- Μια αιτιολογική σύνδεση είναι μια δομή με τρία πεδία: δύο δείκτες σε ενέργειες του πλάνου, έστω a_p και a_c, και ένα γεγονός p.
  - Το γεγονός p ανήκει τόσο στη λίστα προσθήκης της ενέργειας a_p, όσο και στη λίστα προϋποθέσεων της ενέργειας a_c.
  - Μια αιτιολογική σύνδεση συμβολίζεται ως $a_p \xrightarrow{p} a_c$.
  - Η ενέργεια a_p ονομάζεται *παραγωγός* (*producer*) της σύνδεσης, ενώ η ενέργεια a_c ονομάζεται *καταναλωτής* (*consumer*) της σύνδεσης.
  - Το L είναι το σύνολο όλων των αιτιολογικών συνδέσεων ενός πλάνου.

Αιτιολογικές Συνδέσεις (2/2)

- Μια αιτιολογική σύνδεση δηλώνει ότι κατά το χρονικό διάστημα μεταξύ της εκτέλεσης της ενέργειας a_p και της ενέργειας a_c, το γεγονός p αληθεύει συνεχώς.
  - Για κάθε αιτιολογική σύνδεση της μορφής $a_p \xrightarrow{p} a_c$ του συνόλου L, ο περιορισμός διάταξης $a_p < a_c$ εισάγεται στο σύνολο O

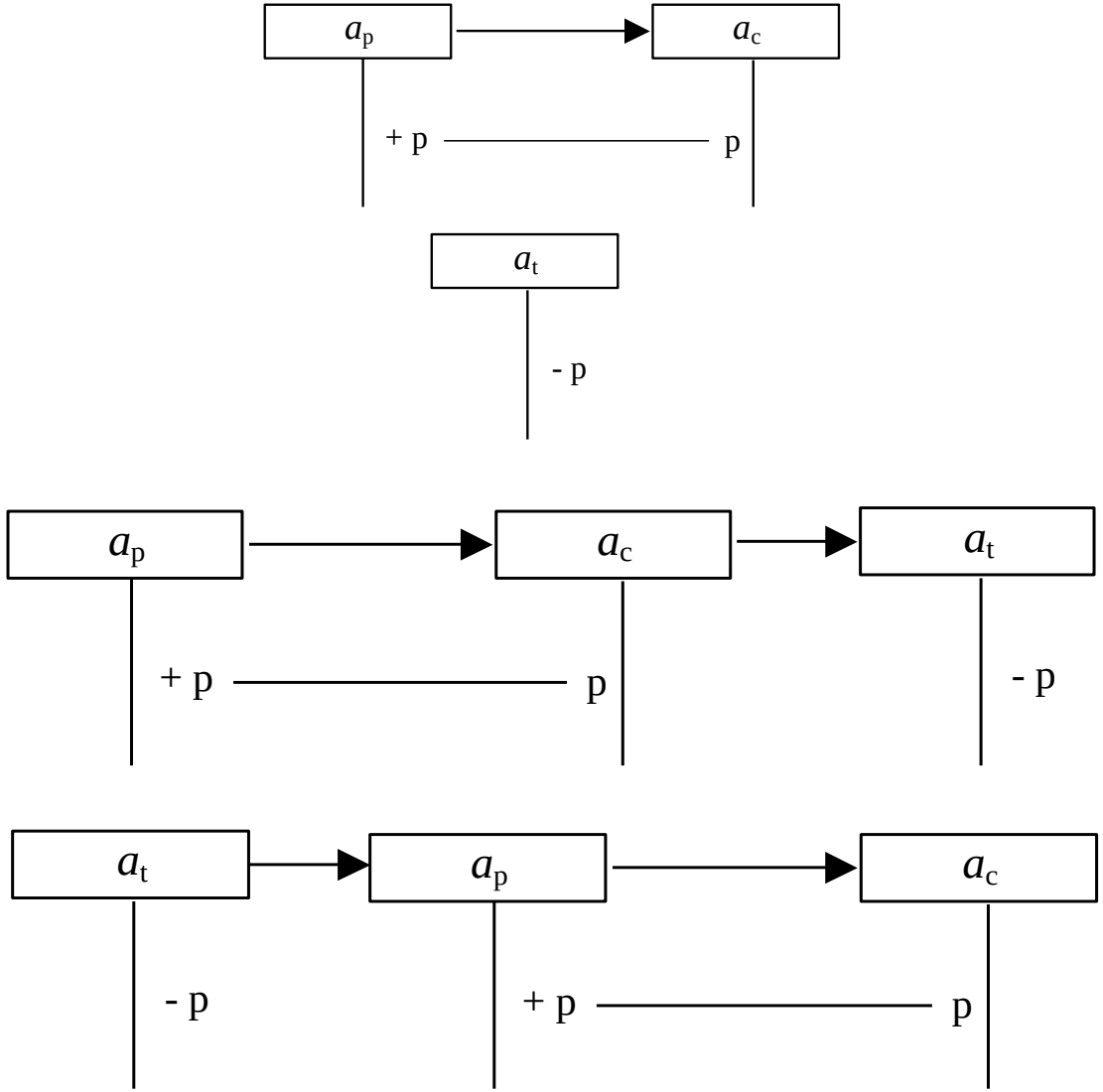
- Διαγραμματική αναπαράσταση αιτιολογικής σύνδεσης



Απειλές (Threats)

- Έστω μια αιτιολογική σύνδεση $a_p \xrightarrow{p} a_c$ και μια τρίτη ενέργεια $a_t \in A$. Λέμε ότι η ενέργεια a_t αποτελεί απειλή για την αιτιολογική σύνδεση $a_p \xrightarrow{p} a_c$, εάν:
  - το σύνολο $O \cup \{a_p < a_t < a_c\}$ είναι συνεπές, και
  - $p \in \text{Del}(a_t)$
- Μια απειλή αντιμετωπίζεται με δύο τρόπους:
  - προβιβασμό (promotion)
  - υποβιβασμό (demotion)

Παράδειγμα απειλής και αντιμετώπισή της με προβιβασμό και υποβιβασμό.



Μη Γραμμικά Πλάνα

- Ένα μη-γραμμικό πλάνο λέγεται *πλήρες* (*complete*), όταν:
  - Κάθε ενέργεια που εμφανίζεται είτε σε κάποια αιτιολογική σύνδεση του συνόλου L ή σε κάποιον περιορισμό διάταξης του συνόλου O, αναφέρεται επίσης στο σύνολο των ενεργειών A.
  - Για κάθε ενέργεια $a \in A$ και για κάθε προϋπόθεση $p \in Pre(a)$, υπάρχει μια αιτιολογική σύνδεση της μορφής $b \xrightarrow{p} a$ στο σύνολο L, όπου $b \in A$.
  - Αν το πλάνο περιέχει μία αιτιολογική σύνδεση $b \xrightarrow{P} a$ και μια ενέργεια c που την απειλεί, στο σύνολο O υπάρχει είτε η διάταξη $c<b$ ή η $a<c$.
- Μία *τοπολογική διάταξη* (*topological sort*) ενός μη-γραμμικού πλάνου είναι μία γραμμική ακολουθία των ενεργειών του, τέτοια ώστε:
  - Η πρώτη ενέργεια στην ακολουθία είναι η **START**.
  - Η τελευταία ενέργεια στην ακολουθία είναι η **FINISH**.
  - Για κάθε αιτιολογική σύνδεση $b \xrightarrow{p} a$, η ενέργεια b προηγείται της ενέργειας a.
  - Για κάθε περιορισμό διάταξης $b<a$ του συνόλου O, η ενέργεια b προηγείται της a.

Αλγόριθμος Παραγωγής Μερικώς Διατεταγμένων Πλάνων

Αλγόριθμος POP((*A*, *O*, *L*), *Agenda*)

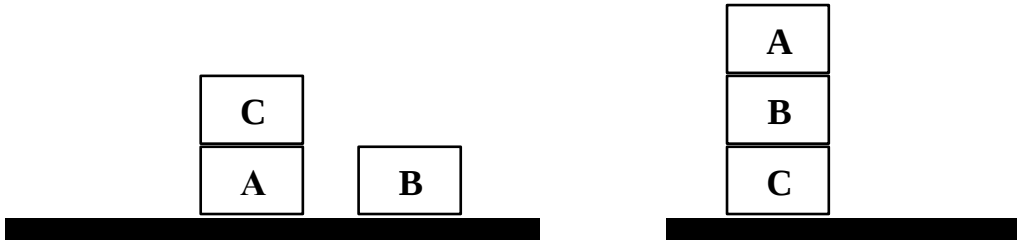
1. Εάν *Agenda*=∅, επέστρεψε το πλάνο (*A*, *O*, *L*).
2. Έστω (q, a_{need}) ένα στοιχείο της *Agenda* (προφανώς ισχύει $a_{need} \in A$ και $q \in Pre(a_{need})$).
3. Έστω a_{add} μια ενέργεια, τέτοια ώστε $q \in Add(a_{add})$. Η ενέργεια αυτή μπορεί είτε να είναι μια από τις ενέργειες του συνόλου *A*, τέτοια ώστε να μπορεί να διαταχθεί χρονικά πριν από την a_{need}, ή να είναι μια νέα ενέργεια. Εάν δεν υπάρχει τέτοια ενέργεια, τότε επέστρεψε αποτυχία. Θέσε $L'=L \cup \{a_{add} \xrightarrow{q} a_{need}\}$, $O'=O \cup \{a_{add} < a_{need}\}$. Εάν η ενέργεια a_{add} είναι μια νέα ενέργεια, τότε $A'=A \cup \{a_{add}\}$ και $O'=O \cup \{\text{START} < a_{add} < \text{FINISH}\}$, ειδάλλως $A'=A$.
4. Θέσε $Agenda'=Agenda-\{\langle q, a_{need}\rangle\}$. Εάν η ενέργεια a_{add} ήταν μια νέα ενέργεια, τότε για κάθε $q_i \in Pre(a_{add})$ πρόσθεσε το στοιχείο $\langle q_i, a_{add}\rangle$ στην *Agenda*'.
5. Για κάθε ενέργεια $a_t \in A'$, η οποία μπορεί να αποτελέσει απειλή για κάποια αιτιολογική σύνδεση $a_p \xrightarrow{q} a_c \in L'$, υπολόγισε το νέο σύνολο περιορισμών διάταξης *O*', επιλέγοντας μια από τις παρακάτω δύο σχέσεις, ελέγχοντας ώστε το σύνολο *O*' να είναι συνεπές:

   $$O' = O \cup \{a_t < a_p\}$$
   $$O' = O \cup \{a_c < a_t\}$$

   Εάν σε καμία από τις παραπάνω δύο περιπτώσεις το σύνολο *O*' είναι συνεπές, επέστρεψε αποτυχία.
6. POP((*A*', *O*', *L*'), *Agenda*')

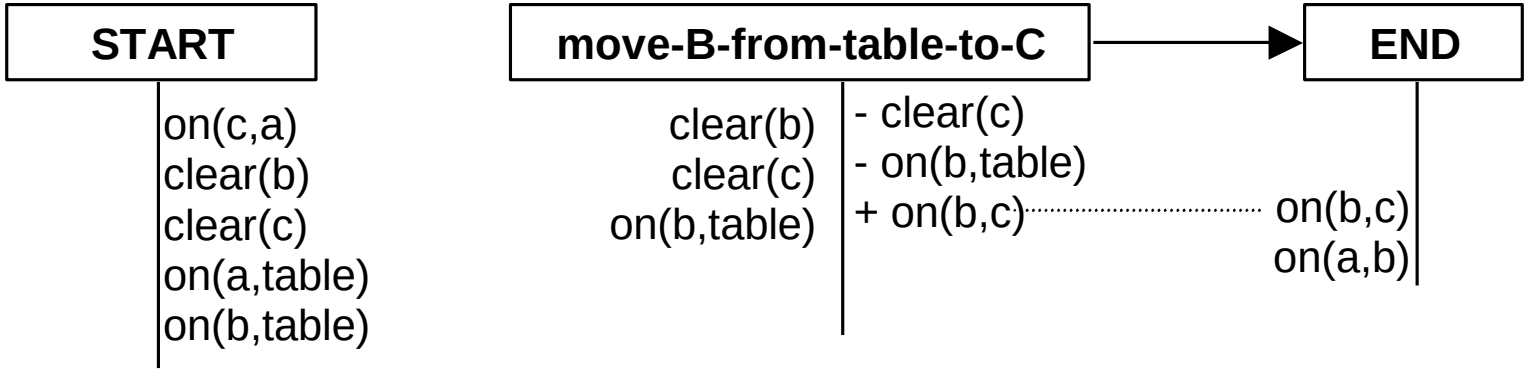
Παράδειγμα (1/3)

- Πρόβλημα



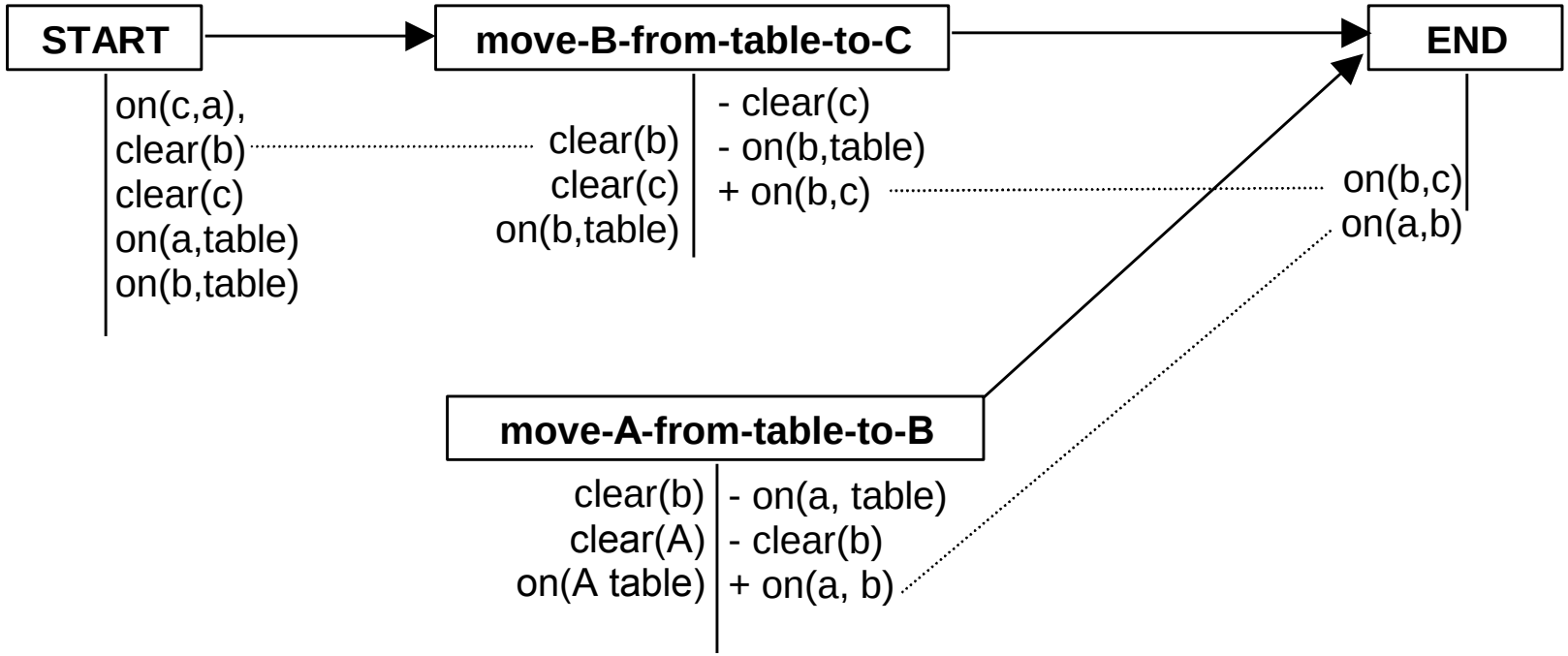
Αρχική κατάσταση Τελική κατάσταση

- Μερικό πλάνο με μια ενέργεια.



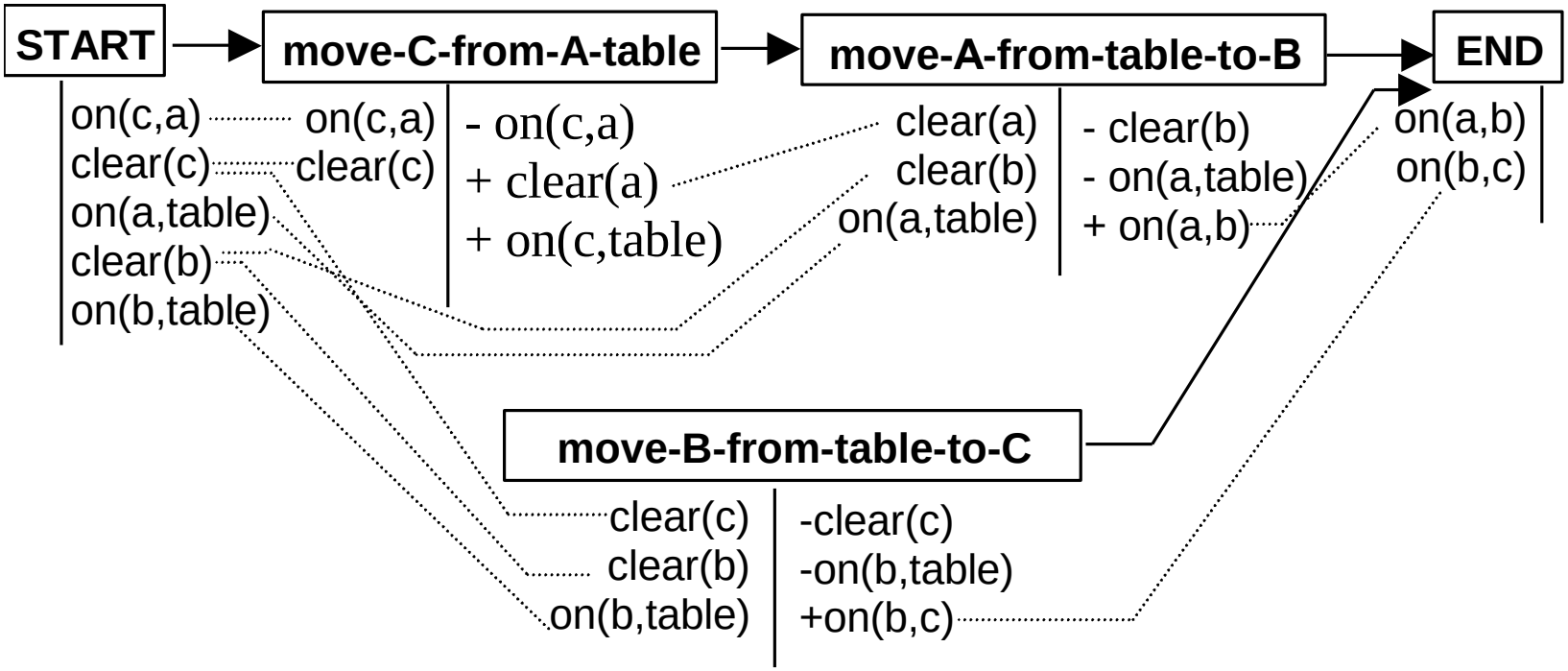
Παράδειγμα (2/3)

- Μερικό πλάνο με δύο ενέργειες.



Παράδειγμα (3/3)

- Ενδεχόμενο τελικό πλάνο



Χρήση Μη-Συγκεκριμένων Ενεργειών

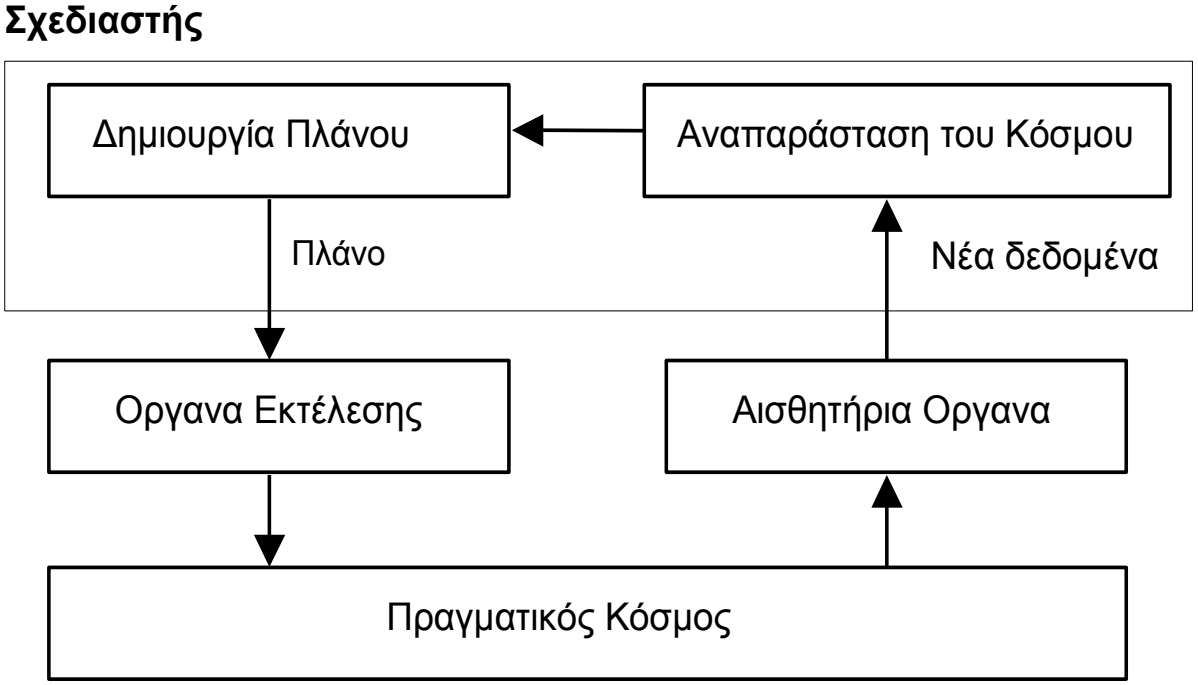
- Κατά την επιλογή μιας ενέργειας δεσμεύονται μόνο οι μεταβλητές αυτής που είναι απαραίτητες για την ενοποίηση ενός γεγονότος που η ενέργεια παράγει με την αντίστοιχη μη υποστηριζόμενη προϋπόθεση μιας άλλης ενέργειας.
- Επιτρέπονται δύο είδη περιορισμών δέσμευσης (binding constraints):
  - Οι περιορισμοί κοινού προσδιορισμού (codesignation constraints) της μορφής ?x=?y ή ?x=c.
  - Οι περιορισμοί μη-κοινού προσδιορισμού (non-codesignation constraints), της μορφής ?x≠?y ή ?x≠c.
- Τα προβλήματα ορίζονται ως μια τετράδα συνόλων (*A*, *O*, *L*, *B*), όπου:
  - Το σύνολο *A* περιέχει σχήματα ενεργειών
  - Το σύνολο *B* περιέχει περιορισμούς δέσμευσης.
- Οι απειλές μπορούν να αντιμετωπίζονται με την εισαγωγή στο σύνολο *B* κατάλληλων περιορισμών μη-κοινού προσδιορισμού.

Εκτέλεση Πλάνων

- Ένας σχεδιαστής (planner) αναλαμβάνει τη δημιουργία ενός πλάνου έχοντας σαν βάση την αναπαράσταση του κόσμου σε μία δεδομένη χρονική στιγμή, όπως την έχει δημιουργήσει μέσω κάποιων αισθητήρων (***sensors***).
- Το πλάνο μετά τη δημιουργία του μεταβιβάζεται στα απαραίτητα όργανα εκτέλεσης (***effectors***).
- Κατά τη διάρκεια δημιουργίας του πλάνου ο κόσμος δεν παραμένει απαραίτητα ο ίδιος, οπότε πολλές φορές η εκτέλεση ενός πλάνου αντιμετωπίζει προβλήματα.
  - Υπάρχει λοιπόν διαφορά μεταξύ του πραγματικού κόσμου και του μοντέλου του κόσμου στο οποίο βασίζεται ο σχεδιαστής.
- Για να εκτελεστεί μία ενέργεια πρέπει ο πραγματικός κόσμος να ανταποκρίνεται στις προϋποθέσεις της ενέργειας.
  - Εκτέλεση σημαίνει πώς τα αποτελέσματα κάθε ενέργειας (με τη χρονική ακολουθία που εμφανίζονται στο πλάνο) εμφανίζονται στον πραγματικό κόσμο.

Εκτέλεση Πλάνων (συνέχεια)

- Σε περίπτωση αλλαγής του κόσμου ή αποτυχίας εκτέλεσης μιας ενέργειας, υπάρχουν τρεις δυνατότητες:
  - Να επαναληφθεί η δημιουργία του πλάνου με νέα πλέον δεδομένα.
  - Να εκτελεστεί ένα εναλλακτικό πλάνο που είναι εφαρμόσιμο ή το ίδιο πλάνο από διαφορετικό σημείο του.
  - Να γίνει τροποποίηση του πλάνου στα σημεία όπου εμφανίζεται το πρόβλημα



- Κάποια συστήματα ΤΝ που περιέχουν σχεδιαστές πλάνων ονομάζονται πράκτορες σχεδιαστές (planning agents).

Εξελιγμένες Τεχνικές Σχεδιασμού

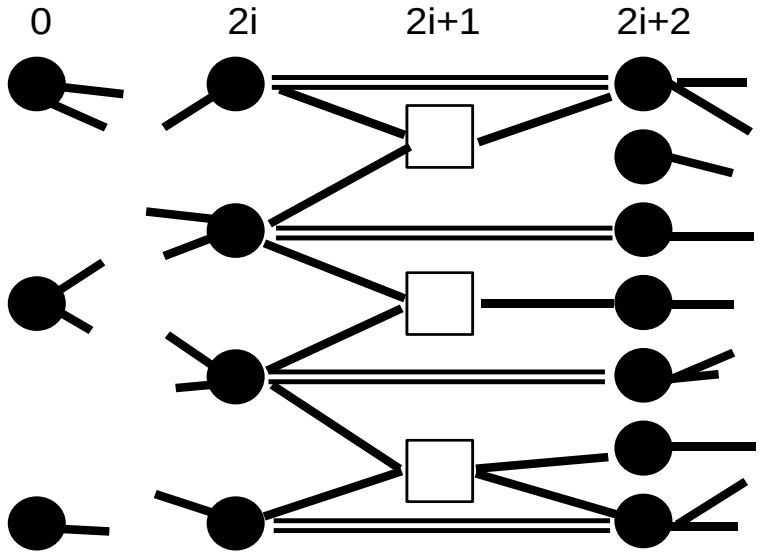
Σχεδιασμός Βασισμένος σε Γράφους

Γράφος σχεδιασμού (1/2)

- Ο γράφος σχεδιασμού αποτελείται από αριθμημένα επίπεδα κόμβων
  - Κόμβοι των γεγονότων ή προτάσεων (fact nodes ή proposition nodes), στα άρτια επίπεδα.
  - Κόμβοι των ενεργειών (action nodes), στα περιττά επίπεδα
- Το επίπεδο με αριθμό 0 περιλαμβάνει έναν κόμβο για κάθε γεγονός της αρχικής κατάστασης.
- Επαναλαμβανόμενη εναλλαγή δύο φάσεων:
  - *Επέκταση του γράφου* (*graph expansion*): Επεκτείνει προοδευτικά στο χρόνο το *γράφο σχεδιασμού* (*planning graph*), μέχρις ότου είτε βρεθεί ένα πλάνο-λύση ή ικανοποιηθεί μια αναγκαία (αλλά όχι ικανή) συνθήκη για τη μη-ύπαρξη πλάνου.
  - *Εξαγωγή λύσης* (*solution extraction*): Αναζήτηση με κατεύθυνση προς-τα-πίσω μέσα στο γράφο σχεδιασμού, με σκοπό την εύρεση ενός πλάνου-λύσης του προβλήματος.

Γράφος σχεδιασμού (2/2)

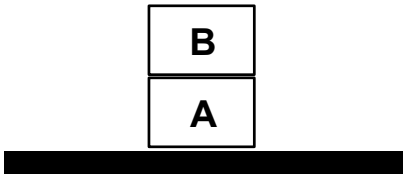
- Οι ακμές συνδέουν:
  - Τα γεγονότα ενός επιπέδου με τις ενέργειες του επόμενου επιπέδου που τα έχουν ως προϋποθέσεις.
  - Τις ενέργειες ενός επιπέδου με τα γεγονότα των λιστών προσθήκης αυτών στο επόμενο επίπεδο.
- Ενέργειες διατήρησης
  - Συμβολίζονται με `noop` (no-operator)



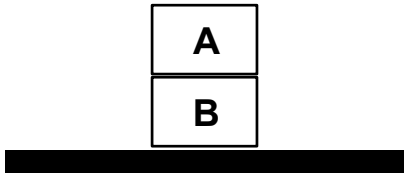
Σχέσεις Αμοιβαίου Αποκλεισμού

- Μια σχέση αμοιβαίου αποκλεισμού αναφέρεται πάντα σε δύο κόμβους του ίδιου επιπέδου και δηλώνει ότι αυτοί δεν μπορούν να βρίσκονται ταυτόχρονα στο ίδιο έγκυρο πλάνο.
  - Δύο ενέργειες a_1 και a_2 στο επίπεδο i είναι αμοιβαία αποκλειόμενες, εάν:
    - Υπάρχουν γεγονότα που εμφανίζονται στη λίστα προσθήκης της μιας και στη λίστα διαγραφής της άλλης (αντιφατικά αποτελέσματα - inconsistent effects), δηλαδή $Add(a_1) \cap Del(a_2) \neq \varnothing \vee Add(a_2) \cap Del(a_1) \neq \varnothing$.
    - Υπάρχει γεγονός που εμφανίζεται στη λίστα προϋποθέσεων της μιας ενέργειας και στη λίστα διαγραφής της άλλης (παρέμβαση – interference), δηλαδή $Pre(a_1) \cap Del(a_2) \neq \varnothing \vee Pre(a_2) \cap Del(a_1) \neq \varnothing$.
    - Υπάρχουν δύο γεγονότα p και q, τέτοια ώστε $p \in Pre(a_1)$ και $q \in Pre(a_1)$, τα οποία είναι αμοιβαία αποκλειόμενα (ανταγωνιστικές απαιτήσεις – competing needs).
  - Δύο γεγονότα στο επίπεδο i είναι αμοιβαία αποκλειόμενα, εάν όλες οι ενέργειες στο επίπεδο i-1, συμπεριλαμβανομένων των ενεργειών noop, που επιτυγχάνουν αυτά τα γεγονότα είναι μεταξύ τους αμοιβαίως αποκλειόμενες (ασύμβατη υποστήριξη – inconsistent support).

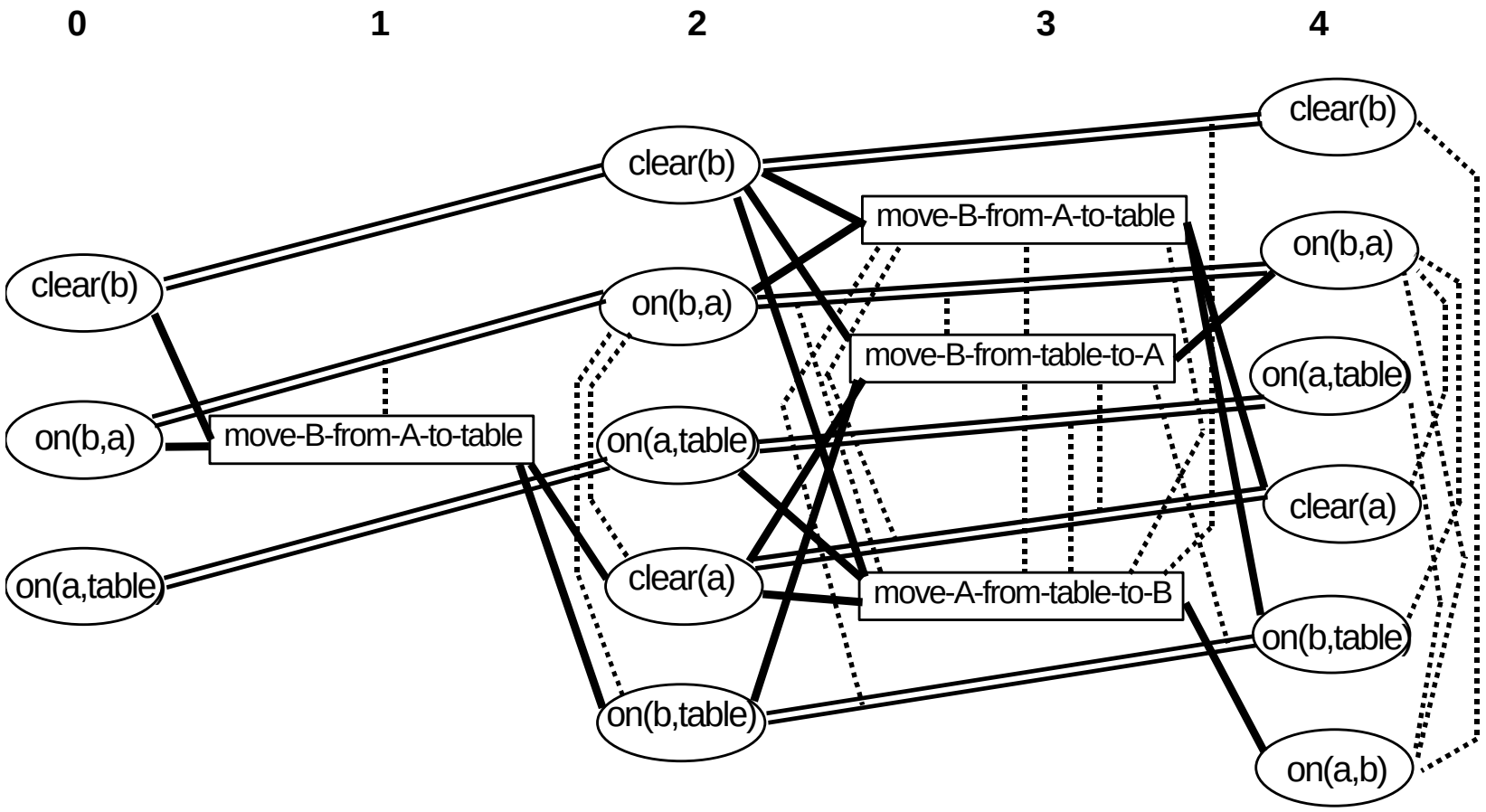
Παράδειγμα



Αρχική κατάσταση



Τελική κατάσταση

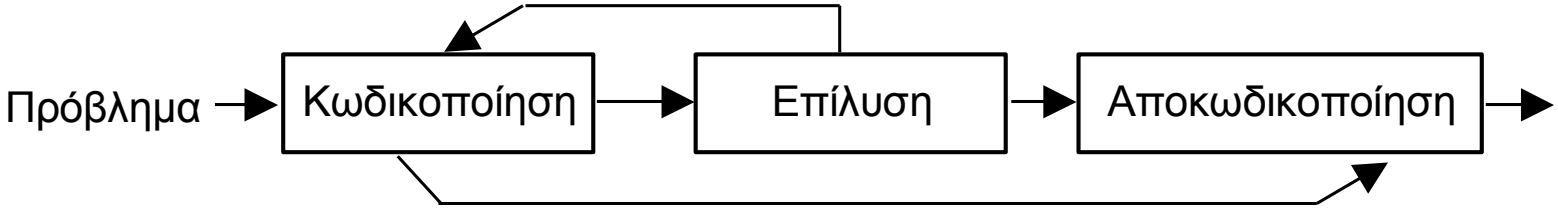


Σχεδιασμός Βασισμένος σε Γράφους

Εξαγωγή λύσης

- ❖ Ξεκινά μόλις σε κάποιο επίπεδο γεγονότων i εμφανιστούν όλα τα γεγονότα των στόχων, χωρίς καμιά σχέση αμοιβαίου αποκλεισμού μεταξύ τους.
  - ❑ Αναγκαία αλλά όχι ικανή συνθήκη για την ύπαρξη πλάνου.
- ❖ Τα γεγονότα των στόχων πρέπει να υποστηριχθούν από μη αμοιβαία αποκλειόμενες ενέργειες του προηγούμενου επιπέδου.
- ❖ Αναδρομικά, οι προϋποθέσεις των ενεργειών αυτών πρέπει να υποστηριχθούν από μη αμοιβαία αποκλειόμενες ενέργειες του προηγούμενού τους επιπέδου, μέχρι να φθάσουμε στο πρώτο επίπεδο.
- ❖ Εάν δεν βρεθεί τέτοιο πλάνο, ο γράφος επεκτείνεται κατά 2 ακόμη επίπεδα και η διαδικασία επαναλαμβάνεται.
- ❖ Συνθήκη τερματισμού είναι η εύρεση δύο εντελώς ίδιων επιπέδων γεγονότων.

Σχεδιασμός με Ικανοποίηση Προτάσεων



- Υπόθεση σχετικά με τον αριθμό των βημάτων του πλάνου-λύσης.
- Κωδικοποίηση σαν πρόβλημα ικανοποίησης προτάσεων σε μορφή *σύζευξης διαζεύξεων* (*conjunctive normal form*, CNF).
- Επίλυση με στοχαστικές ή συστηματικές μεθόδους.
- Εάν δεν βρεθεί λύση, επαναλαμβάνεται η διαδικασία για μεγαλύτερο αριθμό βημάτων.

Κωδικοποίηση (1/2)

- Η σύζευξη των γεγονότων της αρχικής κατάστασης πρέπει να αληθεύει.

$$\mathbf{on(b,a)_0 \wedge on(a,table)_0 \wedge clear(b)_0}$$

- Η σύζευξη των γεγονότων των στόχων πρέπει επίσης να αληθεύει.

$$\mathbf{on(a,b)_4 \wedge on(b,table)_4 \wedge clear(a)_4}$$

- Οι ενέργειες συνεπάγονται τις προϋποθέσεις τους και τα αποτελέσματά τους.

$$\mathbf{move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \Rightarrow on(a,table)_2 \wedge clear(a)_2 \wedge clear(b)_2 \wedge on(a,b)_4 \wedge}$$
$$\mathbf{clear(a)_4 \wedge \neg on(a,table)_4 \wedge \neg clear(b)_4}$$

  - ή ισοδύναμα σε μορφή CNF

$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee on(a,table)_2) \wedge}$$
$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee clear(a)_2) \wedge}$$
$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee clear(b)_2) \wedge}$$
$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee on(a,b)_4) \wedge}$$
$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee clear\ A)_4) \wedge}$$
$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee \neg on(a,table)_4) \wedge}$$
$$\mathbf{(\neg move\text{-}A\text{-}from\text{-}table\text{-}to\text{-}B_3 \vee \neg clear(b)_4)}$$

Κωδικοποίηση (2/2)

- Ενέργειες ενός επιπέδου που είναι αμοιβαία αποκλειόμενες μεταξύ τους δεν μπορούν να εκτελεστούν ταυτόχρονα.

$$\neg\textbf{move-A-from-table-to-B}_3 \vee \neg\textbf{move-B-from-table-to-A}_3$$

- Κάθε γεγονός ενός επιπέδου (εκτός του επιπέδου 0) συνεπάγεται τη διάζευξη όλων των ενεργειών του προηγούμενου επιπέδου που το επιτυγχάνουν (συμπεριλαμβανομένων των ενεργειών διατήρησης).

$$\textbf{on(b,a)}_4 \Rightarrow \textbf{move-B-from-table-to-A}_3 \vee \textbf{(noop on(b,a))}_3$$

  - ή ισοδύναμα σε μορφή CNF

$$\neg\textbf{on(b,a)}_4 \vee \textbf{move-B-from-table-to-A}_3 \vee \textbf{(noop on(b,a))}_3$$

Συστηματική Επίλυση Προβλημάτων

Αλγόριθμος DPLL (CNF έκφραση φ)
Εάν η φ είναι κενή, επέστρεψε αληθές,
αλλιώς εάν υπάρχει πρόταση στη φ που να αποτιμάται ψευδής, επέστρεψε ψευδές,
αλλιώς εάν υπάρχει μια καθαρή μεταβλητή X στη φ, επέστρεψε DPLL($\varphi(X)$),
αλλιώς εάν υπάρχει μια μοναδιαία πρόταση $\{X\}$ στη φ, επέστρεψε DPLL($\varphi(X)$),
αλλιώς

επέλεξε μια μεταβλητή X που εμφανίζεται στη φ,
Εάν DPLL(φ(X))=αληθές, επέστρεψε αληθές,
αλλιώς επέστρεψε DPLL($\varphi(\neg X)$).

Στοχαστική Επίλυση Προβλημάτων

Αλγόριθμος GSAT (CNF έκφραση φ, integer: N_{restarts}, N_{flips})
Από i=1 μέχρι i=N_{restarts}
    Έστω A μια τυχαία ανάθεση τιμών σε όλες τις μεταβλητές της φ.
    Από j=1 μέχρι i=N_{flips}
        Εάν η ανάθεση A ικανοποιεί την φ, επέστρεψε αληθές
        Αλλιώς
            Έστω X η μεταβλητή εκείνη της φ, της οποίας η αντιστροφή της τιμής δίνει το μεγαλύτερο αριθμό ικανοποιημένων προτάσεων στην πρόταση φ (σε περίπτωση ύπαρξης πολλών τέτοιων μεταβλητών, επέλεξε μια τυχαία)
            Τροποποίησε την A, αντιστρέφοντας την τιμή της μεταβλητής X.
Επέστρεψε ψευδές.

Μελέτη Περιπτώσεων Συστημάτων Σχεδιασμού

Εισαγωγή

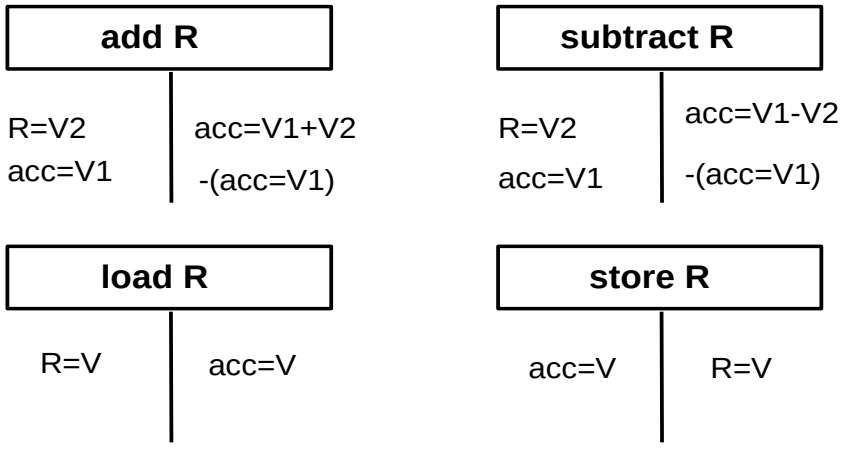
- Παρουσιάστηκαν οι βασικότερες προσεγγίσεις στο πρόβλημα του σχεδιασμού.
  - Το μοντέλο περιγραφής των προβλημάτων σχεδιασμού το οποίο υιοθετήθηκε είναι το STRIPS μοντέλο, το οποίο αν και είναι απλό, είναι αρκετά δημοφιλές.
- Ωστόσο σε πολλές περιπτώσεις χρησιμοποιούνται πιο εκφραστικές γλώσσες περιγραφής των ενεργειών των προβλημάτων, οι οποίες υποστηρίζουν:
  - πόρους (resources),
  - αρνητικές προϋποθέσεις (negative preconditions),
  - αποτελέσματα υπό προϋπόθεση (conditional effects),
  - διάζευξη στις προϋποθέσεις (disjunctive preconditions), αλλά και
  - καθολική ποσοτικοποίηση (universal quantification) στα αποτελέσματα των ενεργειών.
- Δεν εξετάστηκαν επίσης πολλά άλλα θέματα που αφορούν το σχεδιασμό, όπως:
  - ο ιεραρχικός σχεδιασμός (hierarchical planning),
  - ο σχεδιασμός βασισμένος σε παραδείγματα (case-based planning),
  - ο σχεδιασμός με πόρους (resource planning), κλπ.
- Στη συνέχεια θα παρουσιασθούν αντιπροσωπευτικά παραδείγματα συστημάτων σχεδιασμού, τα οποία είτε χρησιμοποιούν πιο εκφραστικές γλώσσες περιγραφής ή πιο εξελιγμένες τεχνικές σχεδιασμού.

Γραμμικά Πλάνα με Ανάστροφη Διάσχιση

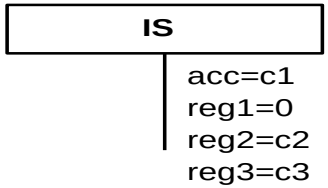
STRIPS

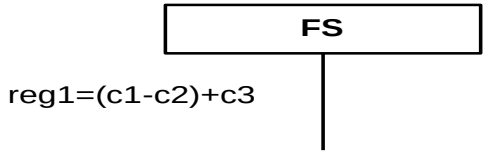
- Όπως έχει ήδη αναφερθεί, ο σχεδιαστής STRIPS παράγει γραμμικά πλάνα με χρήση ανάστροφης διάσχισης στο χώρο των καταστάσεων.
- Παράδειγμα εφαρμογής του σε ένα απλό πρόβλημα σχεδιασμού
  - Έστω ένας πολύ απλός υπολογιστής με ένα συσσωρευτή (Accumulator ή acc) και έναν αριθμό καταχωρητών (Registers ή reg1, reg2, reg3). Οι διαθέσιμες εντολές είναι:
    - add R (πρόσθεσε την τιμή του καταχωρητή R στο συσσωρευτή)
    - subtract R (αφαίρεσε την τιμή του καταχωρητή R από το συσσωρευτή)
    - load R (αντέγραψε την τιμή του καταχωρητή R στο συσσωρευτή)
    - store R (αντέγραψε την τιμή του συσσωρευτή στον καταχωρητή R)

- Οι ενέργειες του προβλήματος του υπολογιστή →

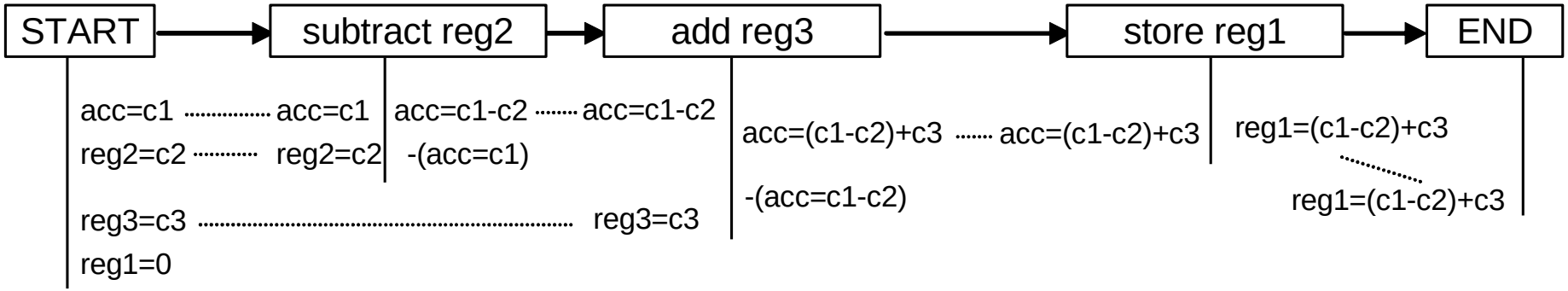


- Αν οι αρχικές τιμές των καταχωρητών και του συσσωρευτή είναι reg1=0, reg2=c2, reg3=c3 και acc=c1, αναζητείται ένα πλάνο το οποίο θα έχει σαν αποτέλεσμα η τιμή (c1-c2)+c3 να υπάρχει στον καταχωρητή reg1.
- Η αρχική (IS) και η τελική (FS) κατάσταση στο πρόβλημα του υπολογιστή:





- Εφαρμόζοντας τον STRIPS, η πρώτη πρόταση που δεν ικανοποιείται είναι η reg1=(c1-c2)+c3.
  - Γίνεται η εισαγωγή στο πλάνο της ενέργειας store reg1, η οποία ικανοποιεί την πρόταση αλλά έχει μία προϋπόθεση, την acc=(c1-c2)+c3, που δεν ικανοποιείται.
  - Στη συνέχεια γίνεται η εισαγωγή νέας ενέργειας, της add reg3, κοκ.
  - Το τελικό πλάνο απεικονίζεται στο σχήμα και είναι η ακολουθία των εντολών:



Ιεραρχικός Σχεδιασμός

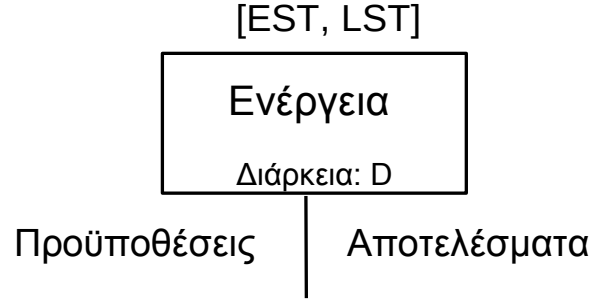
ABSTRIPS (ABstract STRIPS)

- Σε κάθε κατηγόρημα αποδίδεται μία *τιμή σημαντικότητας* (*criticality value*.
- Ο αλγόριθμος του ABSTRIPS είναι ο ίδιος με αυτόν του STRIPS, μόνον που κάθε φορά επαναλαμβάνεται για το υποπρόβλημα που προκύπτει εάν από την αρχική κατάσταση, τους στόχους και τις ενέργειες αφαιρεθούν τα γεγονότα με σημαντικότητα μικρότερη ενός ορίου.
- Σε κάθε επίπεδο σημαντικότητας ο ABSTRIPS δημιουργεί ένα *αφηρημένο πλάνο* (*abstract plan*), το οποίο χρησιμοποιείται ως σκελετός για την κατασκευή του πλάνου του αμέσως πιο λεπτομερούς επιπέδου σημαντικότητας.

Σχεδιασμός με Χρονικές Στιγμές

DEVISER

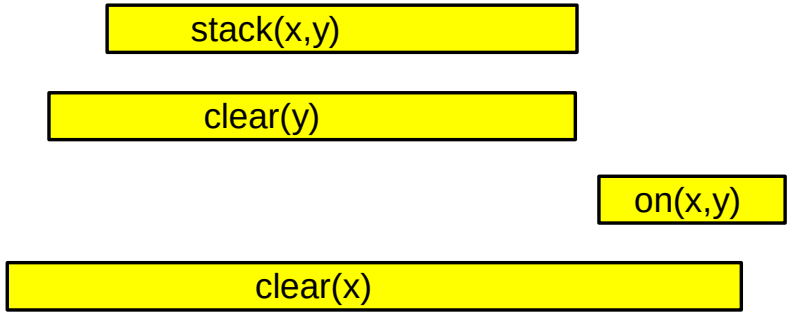
- Σε κάθε ενέργεια αποδίδεται και ένα *χρονικό παράθυρο* [EST, LST] που υποδηλώνει:
  - Το νωρίτερο που μπορεί να αρχίσει η ενέργεια (*earliest starting time - EST*).
  - Το αργότερο που μπορεί να αρχίσει η ενέργεια (*latest starting time - LST*).



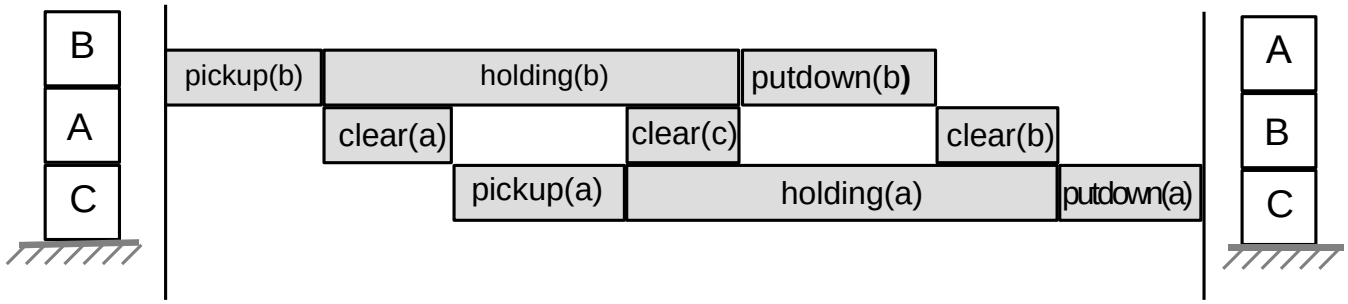
- Για δύο ενέργειες a και b, τέτοιες ώστε η ενέργεια a να πρέπει να εκτελεστεί πριν από την ενέργεια b, οι τιμές EST και LST ανανεώνονται σύμφωνα με τους τύπους:
  - EST(b) = max(EST(b), EST(a) + DUR(a))
  - LST(a) = min(LST(a), LST(b) - DUR(a))
- Σε κάθε βήμα πρέπει να τηρούνται οι εξής περιορισμοί:
  - EST(a) + DUR(a) ≤ EST(b)
  - LST(a) + DUR(a) ≤ LST(b)

Σχεδιασμός με Χρονικά Διαστήματα

- Οι ενέργειες περιέχουν και τους περιορισμούς των χρονικών διαστημάτων κάτω από τους οποίους πρέπει να ισχύουν οι προϋποθέσεις και τα αποτελέσματά τους.
  - Ορισμός ενέργειας stack(x,y)

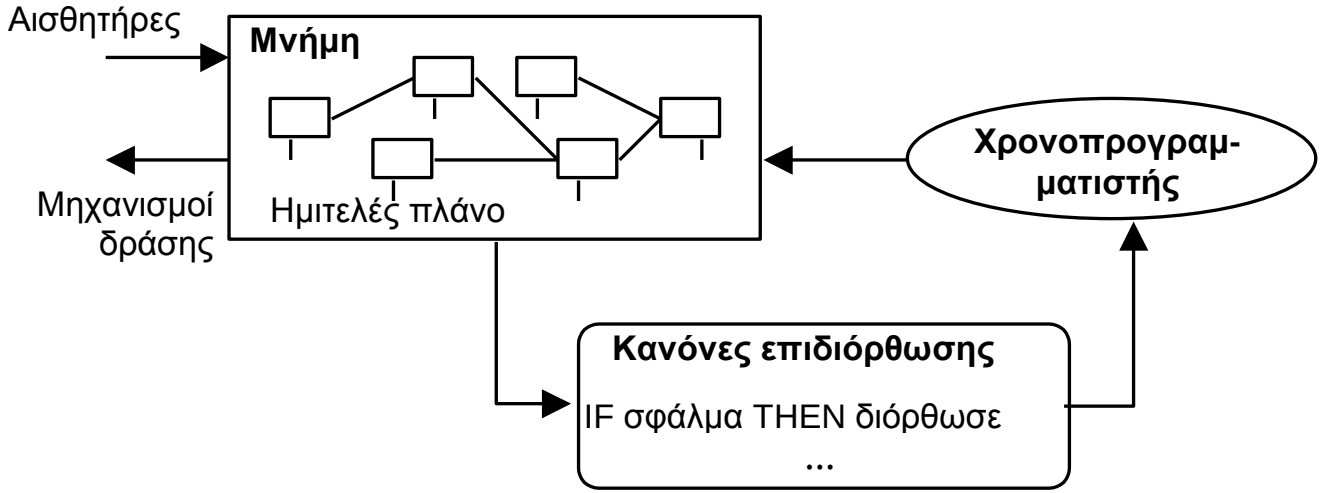


  - Αναπαράσταση πλάνου λύσης



Αλληλεπίδραση Δημιουργίας και Εκτέλεσης Πλάνων

IPEM (Integrated Plan Execution & Monitoring)



- Ο κύκλος λειτουργίας του συστήματος IPEM είναι:
  - Επανέλαβε:
    1) Βρες όλα τα σφάλματα του ημιτελούς πλάνου.
    2) Διάλεξε το σφάλμα με την υψηλότερη προτεραιότητα.
    3) Σύμφωνα με τον αντίστοιχο κανόνα κάνε την επιδιόρθωση.

Ασκήσεις

- 10.1. Ένα εργοστάσιο κατασκευάζει βίδες χρησιμοποιώντας σαν πρώτη ύλη ακατέργαστο σίδηρο. Η διαδικασία κατασκευής μιας βίδας περιλαμβάνει τρία βήματα:
  - α) Επεξεργασία με τόρνο για τη δημιουργία της ελικοειδούς σπείρας.
  - β) Γυάλισμα με τη μηχανή γυαλίσματος.
  - γ) Βαφή (paint).

- Η ακολουθία των παραπάνω βημάτων είναι υποχρεωτική, δηλαδή μία βίδα δεν μπορεί να βαφεί πριν γυαλιστεί κλπ.
- Να γραφούν τρεις τελεστές οι οποίοι να περιγράφουν τις παραπάνω τρεις διαδικασίες.
- Να οριστεί επίσης η αρχική και η τελική κατάσταση, αν θεωρήσουμε ότι έχουμε 3 κομμάτια ακατέργαστου σιδήρου, τα οποία πρέπει να μετατραπούν σε βίδες.
- Για την περιγραφή μπορεί να χρησιμοποιηθεί η γλώσσα Prolog, ή όποια άλλη γλώσσα ή ψευδογλώσσα θέλετε.

❖ 10.2. Έστω ένα πρόβλημα σχεδιασμού που περιγράφεται από τα παρακάτω:
Αρχική κατάσταση $\boldsymbol{I = \{p,q,r,v\}}$, Στόχοι $\boldsymbol{G = \{w,y,z,r,x,b,p\}}$
Ενέργειες:

A_1	
p	w
r	¬p
q	
v	

A_2	
a	x
b	¬a
	v

A_3	
y	a
q	z
	¬q

A_4	
q	b
r	y
v	w
	¬v

❖ Θεωρείστε ότι ο αλγόριθμος αναζήτησης είναι η Αναρρίχηση Λόφων (Hill Climbing) και η ευριστική συνάρτηση δίνεται από τον τύπο $\mathbf{h(S,G) = |G-S|}$, δηλαδή επιστρέφει το πλήθος των στόχων που δεν υπάρχουν στην κατάσταση S.

❖ Να επιλυθεί το παραπάνω πρόβλημα γράφοντας σε κάθε βήμα την τρέχουσα κατάσταση, τις ενέργειες που μπορούν να εφαρμοστούν (με τις τιμές της ευριστικής συνάρτησης) και την ενέργεια που επιλέγεται.
(π.χ. **βήμα 7**: S ={t,g,j}, Ενέργειες: A_8 [5],A_{13} [8],A_{25} [6], Επιλέγεται η A_8)