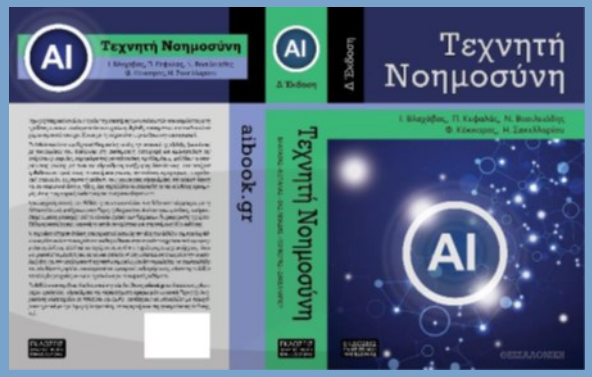
ΜΕΡΟΣ Γ

Σχεδιασμός Ενεργειών (Planning)



Τεχνητή Νοημοσύνη - Δ' Έκδοση-Εκδόσεις Πανεπιστημίου Μακεδονίας
ISBN: 978-618-5196-44-8 - https://aibook.gr/
Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου.

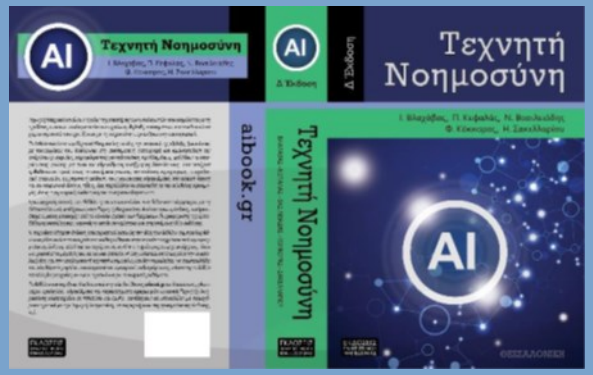
Ι. Βλαχάβας, καθηγητής
Τμήμα Πληροφορικής, ΑΠΘ

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



Τεχνητή Νοημοσύνη - Δ' Έκδοση-Εκδόσεις Πανεπιστημίου Μακεδονίας
ISBN: 978-618-5196-44-8 - https://aibook.gr/
Ι. Βλαχάβας, Π. Κεφαλάς, Ν. Βασιλειάδης, Φ. Κόκκορας, Η. Σακελλαρίου.

Ι. Βλαχάβας, καθηγητής
Τμήμα Πληροφορικής, ΑΠΘ

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. (εξηγ. στο επόμ. slide)

Σχεδιασμός Ενεργειών (Planning) (συνεχ.)

- Οι τυφλοί αλγόριθμοι αναζήτησης δεν μπορούν να διακρίνουν ποιες από τις ενέργειες είναι σχετικές με την επίλυση του προβλήματος.
  - Άμεση συνέπεια αυτού είναι να εφαρμόζονται όλες οι ενέργειες, δημιουργώντας έτσι έναν πολύ μεγάλο χώρο αναζήτησης (φαινόμενο συνδυαστικής έκρηξης).
  - Όμως, ένα μικρό μόνο μέρος αυτού του χώρου είναι σχετικό με την τελική λύση του προβλήματος.
- Οι ευρετικοί αλγόριθμοι αποδίδουν καλύτερα, γιατί εφαρμόζοντας κάθε φορά τις κατάλληλες ενέργειες επικεντρώνουν την αναζήτηση στο χώρο που θα δώσει τελικά τη λύση **αλλά**
  - δεν είναι όμως πάντα εύκολο να βρεθούν ευρετικοί μηχανισμοί ούτε και να κωδικοποιηθούν αυτοί σε συναρτήσεις εξ' αιτίας του τρόπου αναπαράστασης του προβλήματος.
  - Οι συγκεκριμένοι αλγόριθμοι αναζήτησης δεν μπορούν εύκολα να ανάγουν ένα πρόβλημα σε πολλά ανεξάρτητα υποπροβλήματα των οποίων η λύση να βρίσκεται ευκολότερα ή ταχύτερα.
- Απαιτείται ο ορισμός μίας γλώσσας περιγραφής προβλημάτων, που να υποστηρίζει την εφαρμογή αλγορίθμων ικανών να αντιμετωπίσουν επιτυχώς τα παραπάνω ζητήματα.
- Links:
  - ✓ Automated planning and scheduling
  - ✓ A brief overview of AI planning
- Steadtler Planner Game AI Demo

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.
  - θα μπορούσε να χρησιμοποιηθεί:
    - ✓ η προτασιακή λογική (αλλά δεν μπορεί να εκφράσει γενικότητα), ή
    - ✓ κατηγορηματική λογική πρώτης τάξης (δεν μπορεί να περιγράψει ενέργειες με μη προκαθορισμένα αποτελέσματα)
- Γενικά υπάρχει 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 αυτήν, 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.
  - https://en.wikipedia.org/wiki/Shakey_the_robot
  - was the first general-purpose mobile robot to be able to reason about its own actions

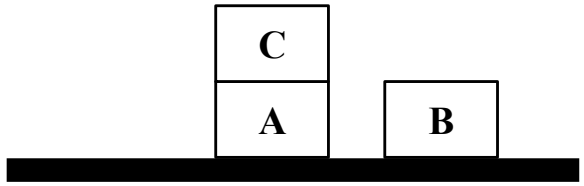
- Το μοντέλο STRIPS γνώρισε μεγάλη απήχηση, κυρίως λόγω της απλότητας και της φυσικότητας του
  - Έχει στοιχεία προτασιακής λογικής και είναι κατάλληλο για προβλήματα όπου δεν εμφανίζεται αβεβαιότητα.
  - Στην αρχική του μορφή δεν υποστήριζε την αναπαράσταση χρονικών και άλλων περιορισμών, ωστόσο στη συνέχεια εμφανίστηκαν πολυάριθμες επεκτάσεις του με πιο πλούσιες εκφραστικές δυνατότητες.

Μοντέλο STRIPS (Παραδοχές)

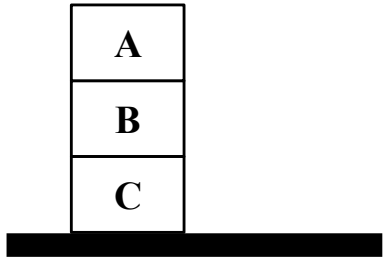
- Στην απλούστερη μορφή του μοντέλου STRIPS γίνονται οι παρακάτω παραδοχές:
  - *Αδιαίρετες ενέργειες* (*indivisible actions*): Δεν ενδιαφέρει η κατάσταση του κόσμου κατά τη διάρκεια εκτέλεσης μιας ενέργειας, παρά μόνο στην αρχή και στο τέλος αυτής.
    - ✓ Επίσης δεν είναι δυνατή η διακοπή της εκτέλεσης 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 κόσμου* (*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:
  - 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.



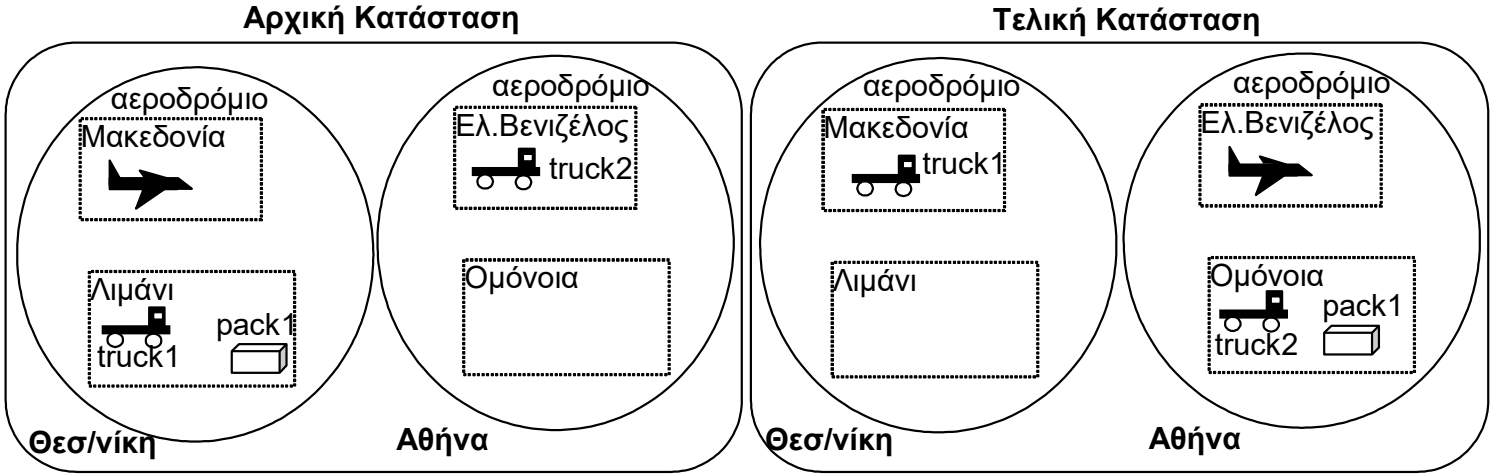
Αρχική κατάσταση



Τελική κατάσταση

  - Σε περιγραφή STRIPS:
    - ✓ **`block(a) ∧ block(b) ∧ block(c) ∧ on(a,table) ∧ on(c,a) ∧ on(b,table) ∧ clear(b) ∧ clear(c)`**
  - Με δεδομένο ότι υπάρχει παντογνωσία της αρχικής κατάστασης, όλα τα γεγονότα τα οποία δεν αναφέρονται σε αυτήν θεωρούνται ότι δεν ισχύουν (είναι ψευδή)
- Τελική κατάσταση:
  - Σε περιγραφή STRIPS: **`on(b,c) ∧ on(a,b)`**
  - Δηλαδή, περιγράφονται ως σύζευξη προτάσεων (στόχων) που πρέπει να αληθεύουν και όχι με πλήρη περιγραφή

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

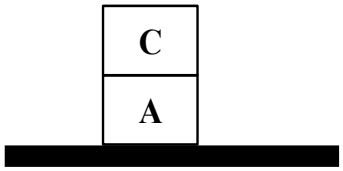
- Ενέργειες (*Actions*) είναι δράσεις που επιφέρουν αλλαγή στην κατάσταση του κόσμου.
  - Για την εφαρμογή των ενεργειών απαιτείται η ικανοποίηση ορισμένων συνθηκών-προϋποθέσεων.
- Στην αναπαράσταση STRIPS, μια *ενέργεια* (action) *a* περιγράφεται με τρεις λίστες γεγονότων:
  - Λίστα προϋποθέσεων (*Precondition 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 αυτή η ενέργεια στη νέα κατάσταση.
  - Λίστα διαγραφής (*Delete list*, Del(a))
    - ✓ Τα γεγονότα της τρέχουσας κατάστασης που δε θα 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(a,table)
on(c,a)



Αρχική κατάσταση

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

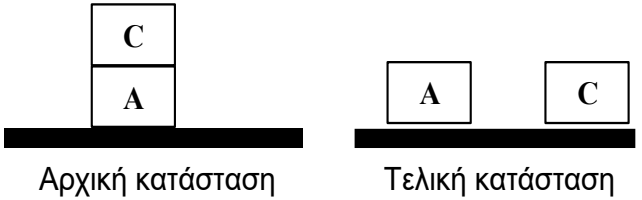
block(a)
block(c)
clear(c)
on(a,table)
~~**on(c,a)**~~

Διαγραφή

- Τελική Κατάσταση

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

Προσθήκη



Αρχική κατάσταση Τελική κατάσταση

Σχήματα Ενεργειών

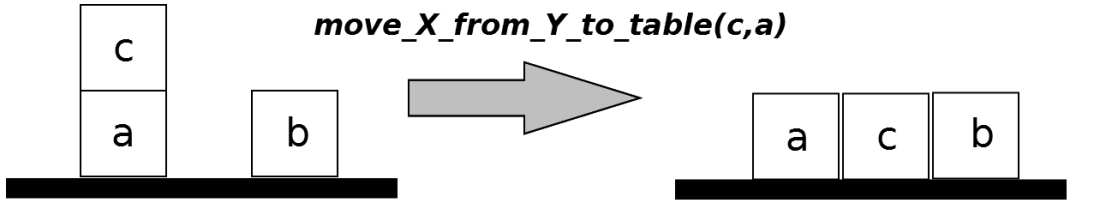
- Επειδή οι ενέργειες ενός προβλήματος σχεδιασμού είναι πάρα πολλές και είναι δύσκολο να απαριθμηθούν, συνηθίζεται να ομαδοποιούνται σε *σχήματα ενεργειών* (*action schemas*) ή *τελεστές* (*operators*).
  - Αυτά μπορεί να γίνουν κανονικές ενέργειες με ανάθεση συγκεκριμένης τιμής στις μεταβλητές τους.
  - 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 (X, Y)** **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



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

block(a)
block(c)
clear(c)
clear(b)
on(b,table)
on(a,table)
~~on(c,a)~~ (Διαγραφή)
clear(a)(Προσθήκη)
on(c,table) (Προσθήκη)

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:

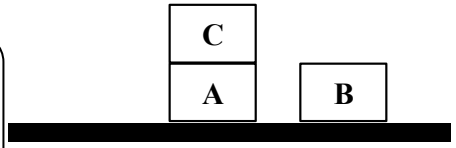
move_C_from_A_to_table

Προϋποθέσεις

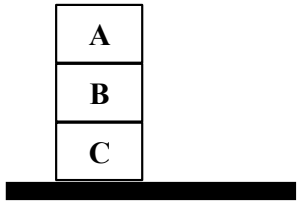
```
block(c)
block(a)
 on(c,a)
clear(c)
```

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

Αποτελέσματα
- Del
+ Add



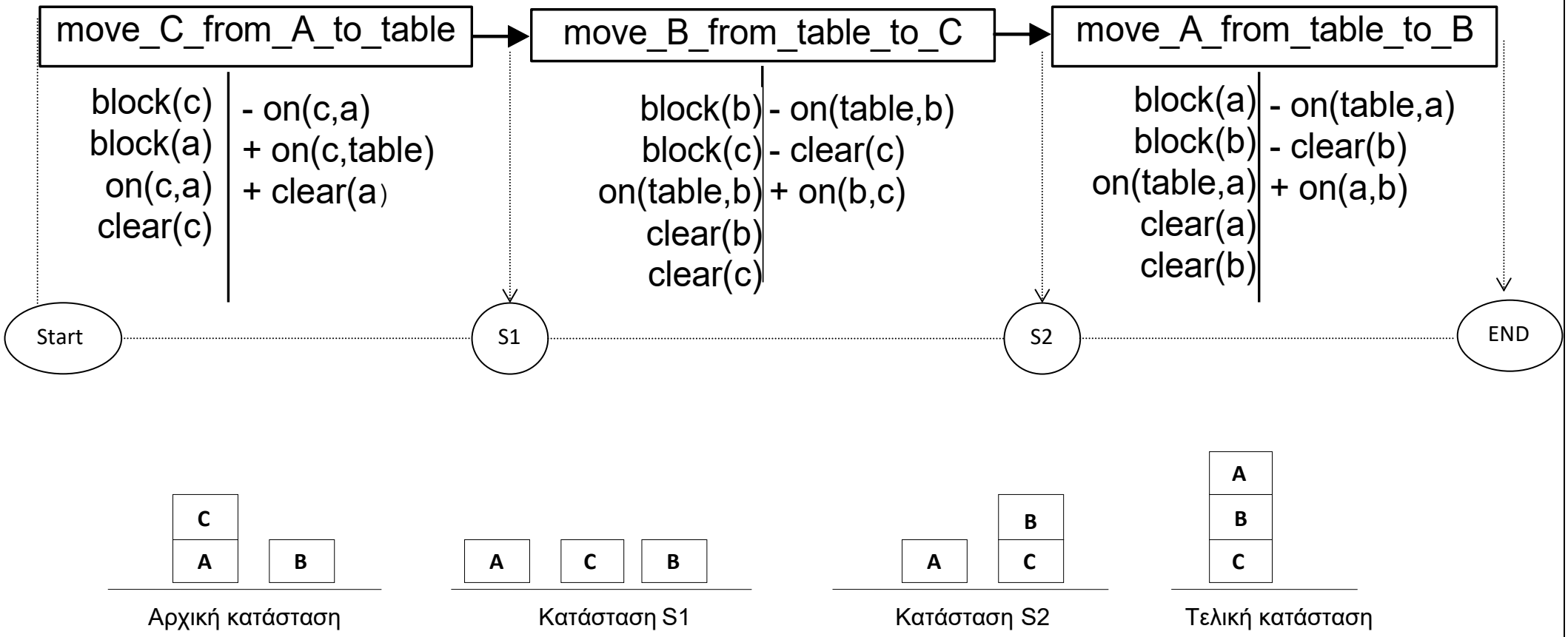
Αρχική κατάσταση



Τελική κατάσταση

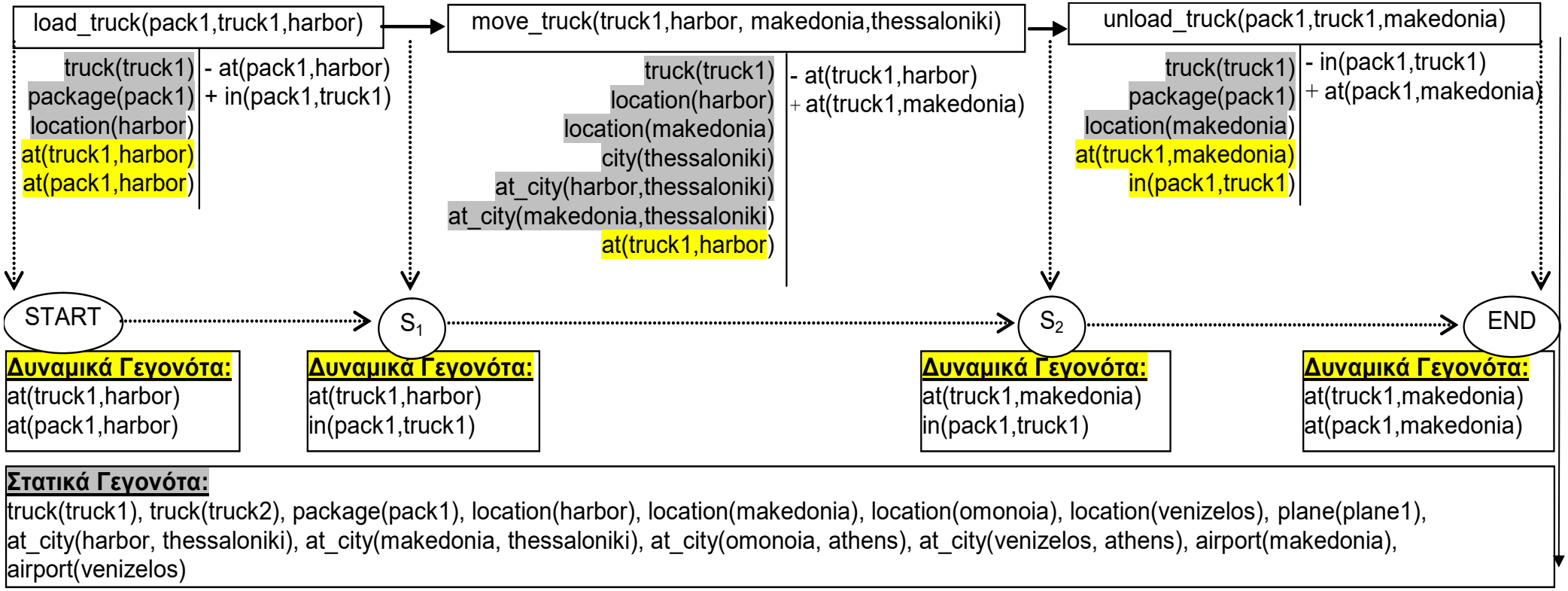
Διαγραμματική Αναπαράσταση πλάνων (1/2)

- Ένα πλάνο μπορεί να αναπαρασταθεί διαγραμματικά με δίκτυο ενεργειών (*procedural network*) όπου οι κόμβοι του είναι οι ενέργειες, ενώ οι (κατευθυνόμενες) ακμές δηλώνουν την ακολουθία των ενεργειών
- π.χ. Λύση του προβλήματος των 3 κύβων



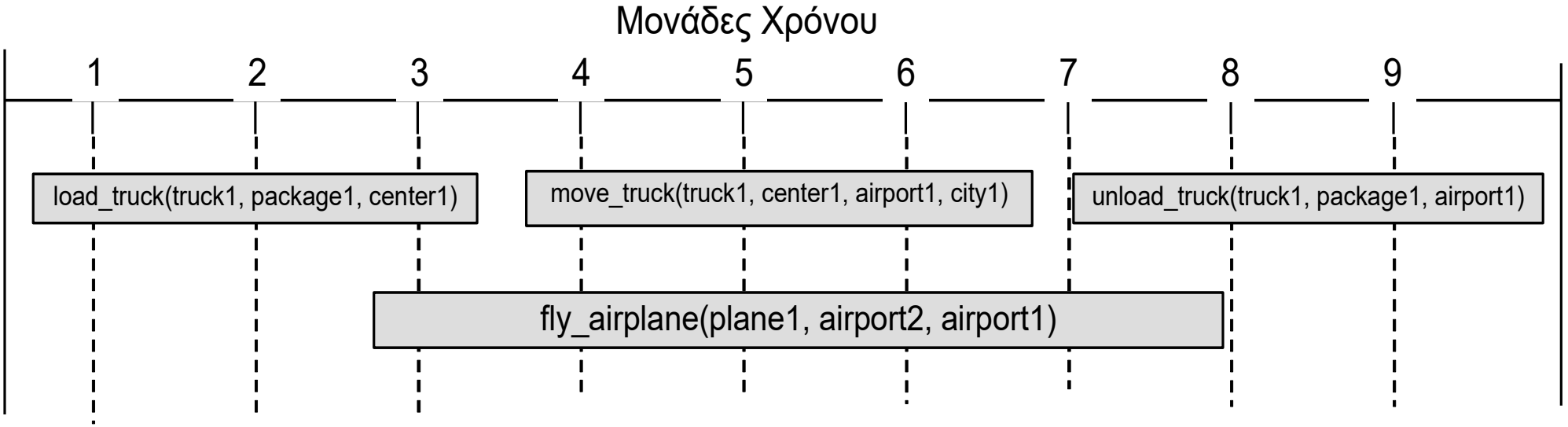
Παράδειγμα: Πλάνα στη μεταφορά φορτίων

- 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 παράλληλης εκτέλεσης ενεργειών, 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 (αλλά μόνο διακριτές).
- Θεωρεί πλήρη βεβαιότητα για την ισχύ των γεγονότων.
  - Π.χ. "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
  - Τα γεγονότα που συμπεριλαμβάνονται στην αναπαράσταση ενός προβλήματος αποτελούν το *πλαίσιό* του (*frame*).
  - Όσο μεγαλύτερο είναι το πλαίσιο ενός προβλήματος, τόσο περισσότερα αξιώματα πλαισίου πρέπει να γραφούν για τις διάφορες ενέργειες.
  - Σαν αποτέλεσμα, υπάρχει τεράστια δυσκολία στην κωδικοποίηση μεγάλων προβλημάτων.
    - ✓ Το πρόβλημα αυτό έμεινε γνωστό σαν το πρόβλημα του πλαισίου (*frame problem*)
  - Το μοντέλο STRIPS παράκαμψε το πρόβλημα δηλώνοντας για κάθε ενέργεια μόνο τα γεγονότα που αυτή αλλάζει και κάνοντας την παραδοχή ότι όλα τα υπόλοιπα γεγονότα παραμένουν ανεπηρέαστα.
  - Αυτό όμως μείωσε σημαντικά την εκφραστική δυνατότητα περιγραφής των προβλημάτων.
    - ✓ Για παράδειγμα, στην περιγραφή της αρχικής κατάστασης δεν δηλώνονται τα χρώματα των κύβων ούτε η εξωτερική θερμοκρασία.

Η γλώσσα αναπαράστασης PDDL

- Η γλώσσα PDDL (*Planning Domain Description Language*) δημιουργήθηκε με σκοπό να υπάρξει ένας ενιαίος συμβολισμός αναπαράστασης προβλημάτων σχεδιασμού
  - Η PDDL έχει αρκετές βελτιωμένες εκδόσεις όπως οι PDDL+, NDDL, MAPL, ADDL, MA-PDDL και άλλες, οι οποίες προσέθεσαν επιπλέον χαρακτηριστικά στην αρχική έκδοση.
- Σε κάθε πρόβλημα ορίζονται:
  - τα κατηγορήματα (predicates) που περιγράφουν ιδιότητες μίας κατάστασης
  - οι ενέργειες (actions) του προβλήματος, με
    - ✓ τις παραμέτρους τους (parameters),
    - ✓ τις προϋποθέσεις τους (preconditions) και
    - ✓ τα αποτελέσματα τους (effects).
  - Οι μεταβλητές (variables) φέρουν το πρόθεμα ?.
  - Οι τελεστές είναι προθεματικοί (prefix), όπως για παράδειγμα ο τελεστής *and*.
- Planning tasks specified in PDDL are separated into two files:
  - A domain file for predicates and actions.
  - A problem file for objects, initial state and goal specification
- Στο Παράρτημα δίνονται 2 παραδείγματα σε PDDL
  - το ένα με το *gripper* το οποίο περιγράφεται και πιο κάτω στην παρουσίαση, και
  - το δεύτερο με 3 παραδείγματα, το gripper, hanoi και logistics

Παράδειγμα των κύβων (blocksworld) σε PDDL

```
(define (domain blocksworld)
  (:requirements :strips)

  (:predicates (clear ?x)
               (on-table ?x)
               (holding ?x)
               (on ?x ?y))

  (:action pickup
           :parameters (?ob)
           :precondition (and (clear ?ob) (on-table ?ob))
           :effect (and (holding ?ob) (not (clear ?ob)) (not (on-table ?ob))))

  (:action putdown
           :parameters (?ob)
           :precondition (and (holding ?ob))
           :effect (and (clear ?ob) (on-table ?ob) (not (holding ?ob))))
```

Οι αρχικές και τελικές καταστάσεις ορίζονται ως εξής:

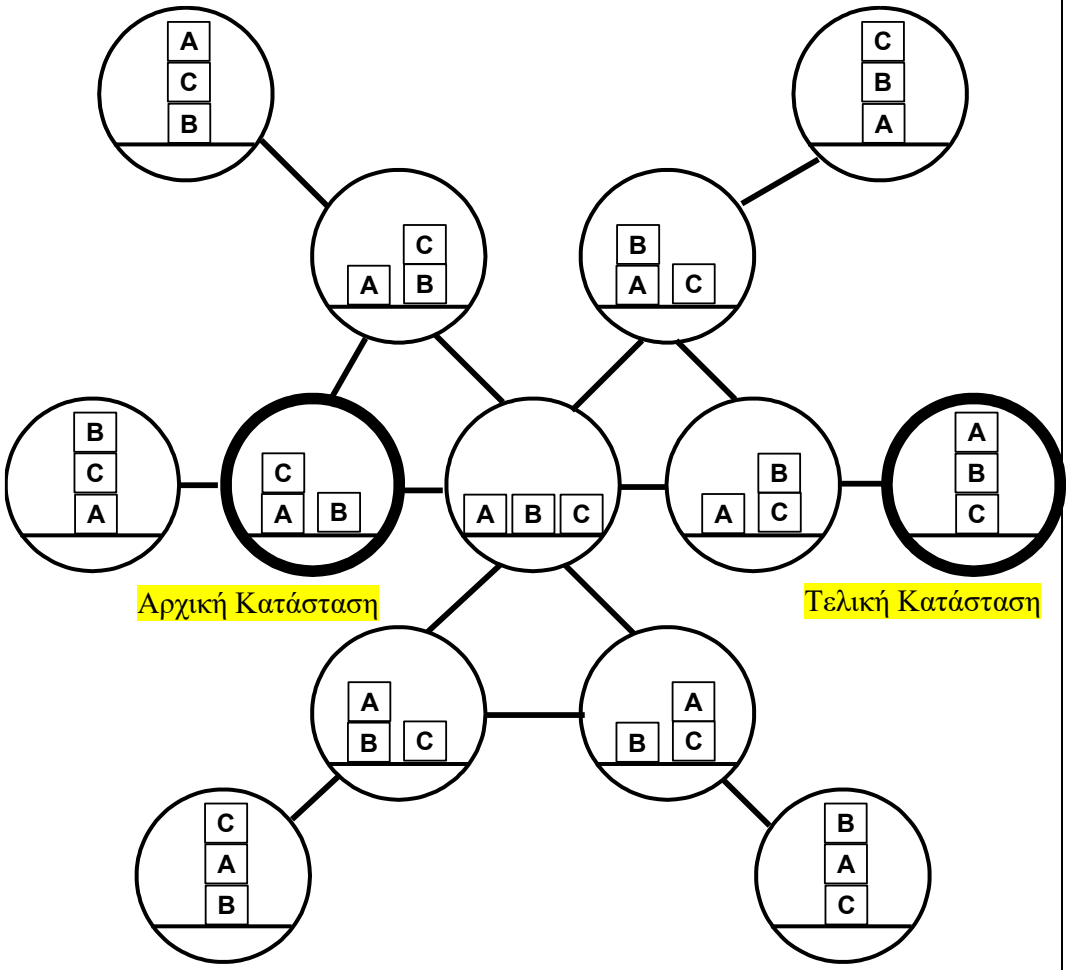
```
(define (problem blocksworld-prob1)
  (:domain blocksworld)
  (:objects a b)
  (:init (on-table a) (on-table b) (clear a) (clear b))
  (:goal (and (on a b))))
```

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 (των 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 (δηλ. των μέσων) που επιτυγχάνουν τους στόχους, πράγμα το οποίο δεν γίνεται εύκολα στην ορθή διάσχιση.

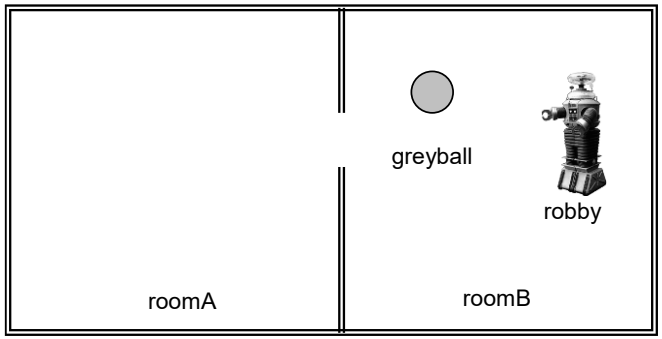
Ανάστροφη Διάσχιση (2/2): Εκτέλεση

- 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.
  - Δηλαδή: $Del(a) \cap G = \varnothing$ και $Add(a) \cap G \neq \varnothing$
- 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 (*gripper*)

- 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)

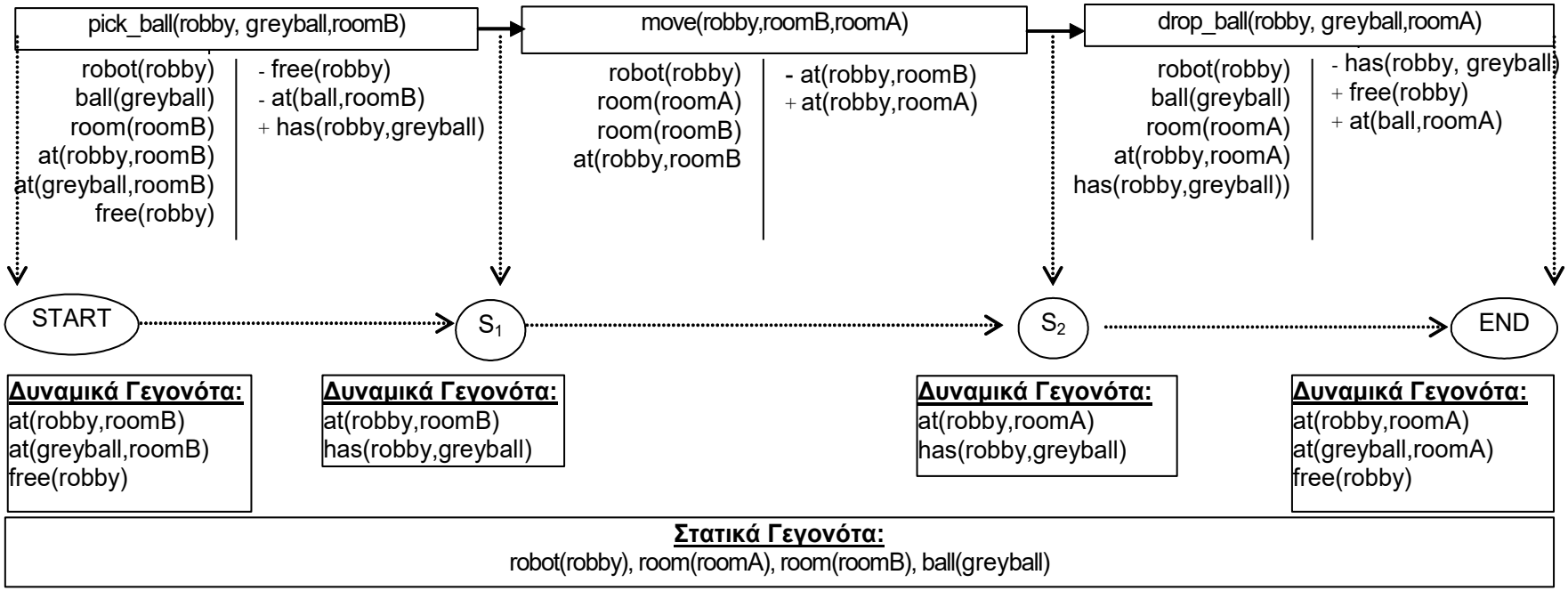
robot(R) ball(B) room(X) at(R,X) has(R,B)	- has(R,B) + at(B,X) + free(R)

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 ενέργεια A_3, 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 $O(b^n)$, 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 $a_2<a_1<a_3$.

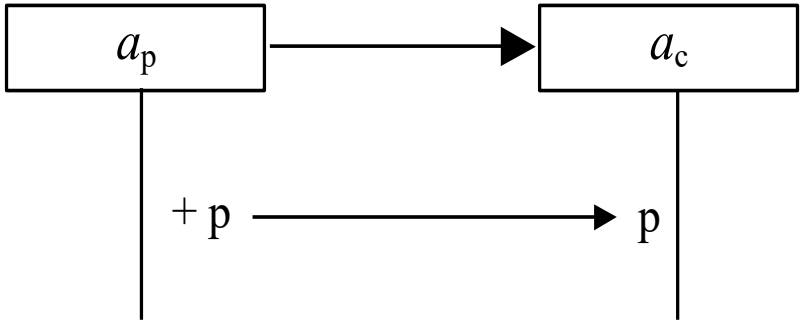
Αιτιολογικές Συνδέσεις (1/2)

- 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*

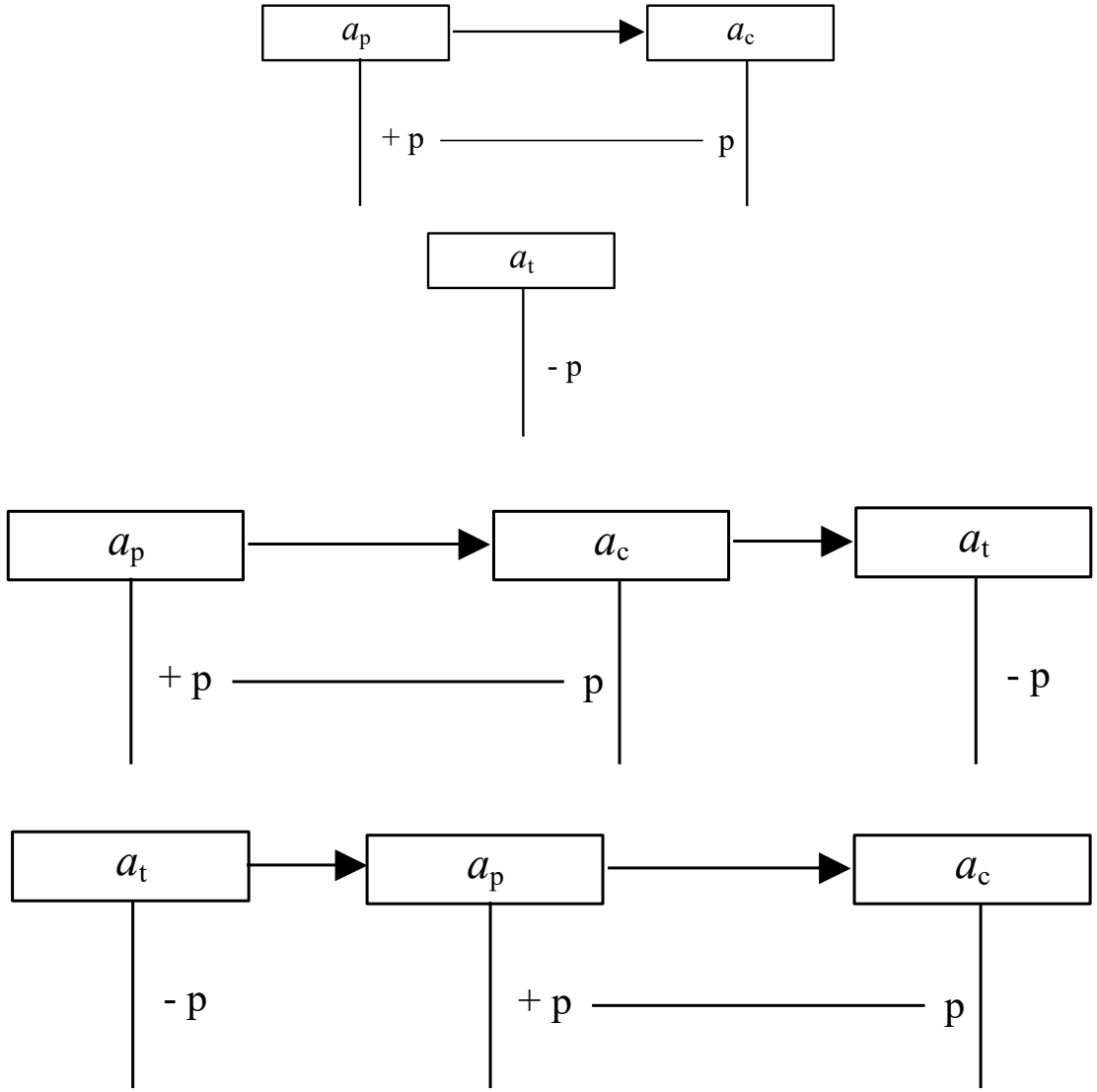
- Διαγραμματική αναπαράσταση αιτιολογικής σύνδεσης



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
  - $p \in Del(a_t)$
- 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 $O'=O \cup \{START<a_{add}<FINISH\}$, 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:

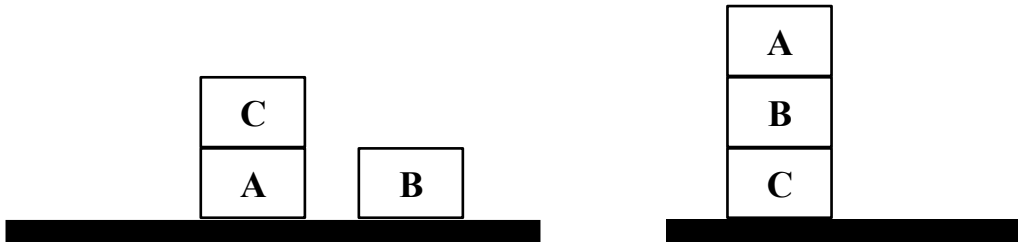
$$O'=O \cup \{a_t<a_p\}$$

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

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

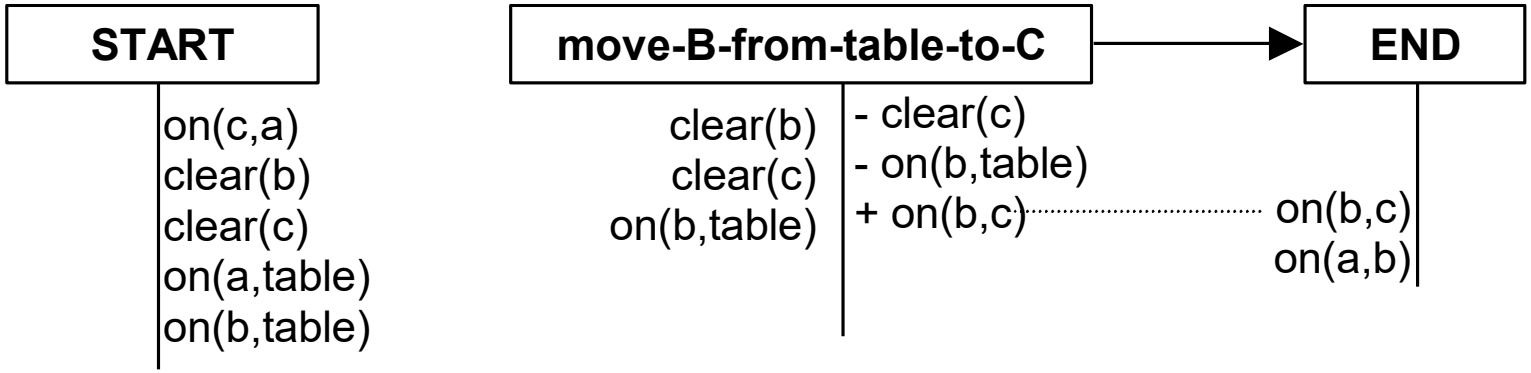
Παράδειγμα (1/3)

- Πρόβλημα



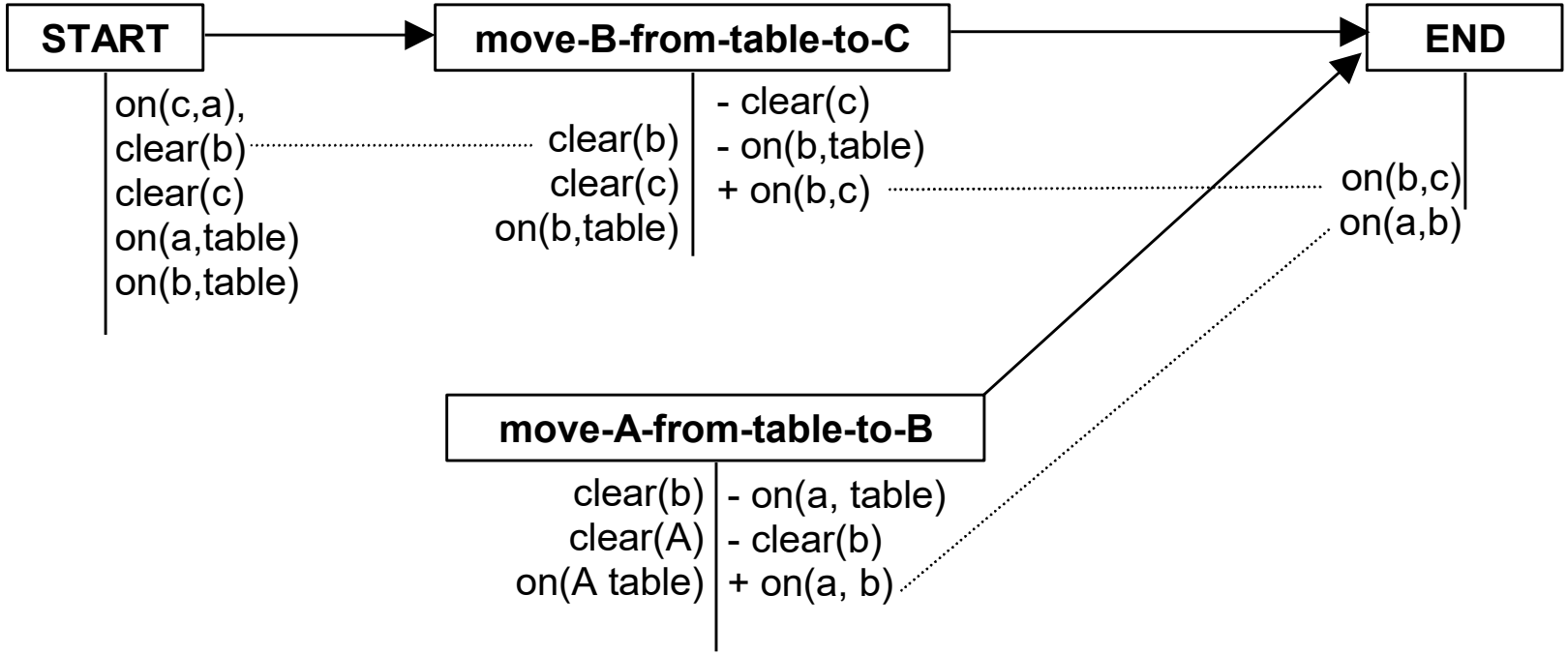
Αρχική κατάσταση Τελική κατάσταση

- Μερικό πλάνο με μια ενέργεια.



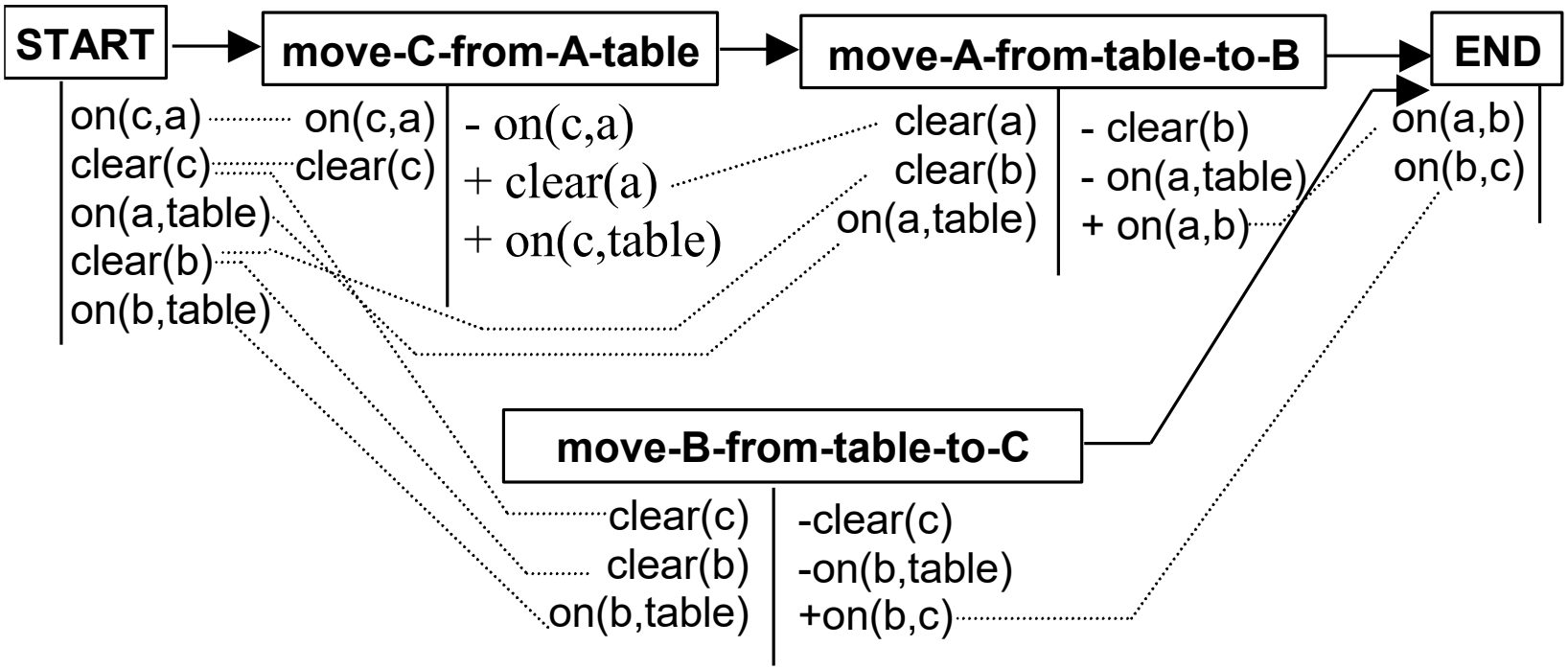
Παράδειγμα (2/3)

- Μερικό πλάνο με δύο ενέργειες.



Παράδειγμα (3/3)

- Ενδεχόμενο τελικό πλάνο



Χρήση Μη-Συγκεκριμένων Ενεργειών

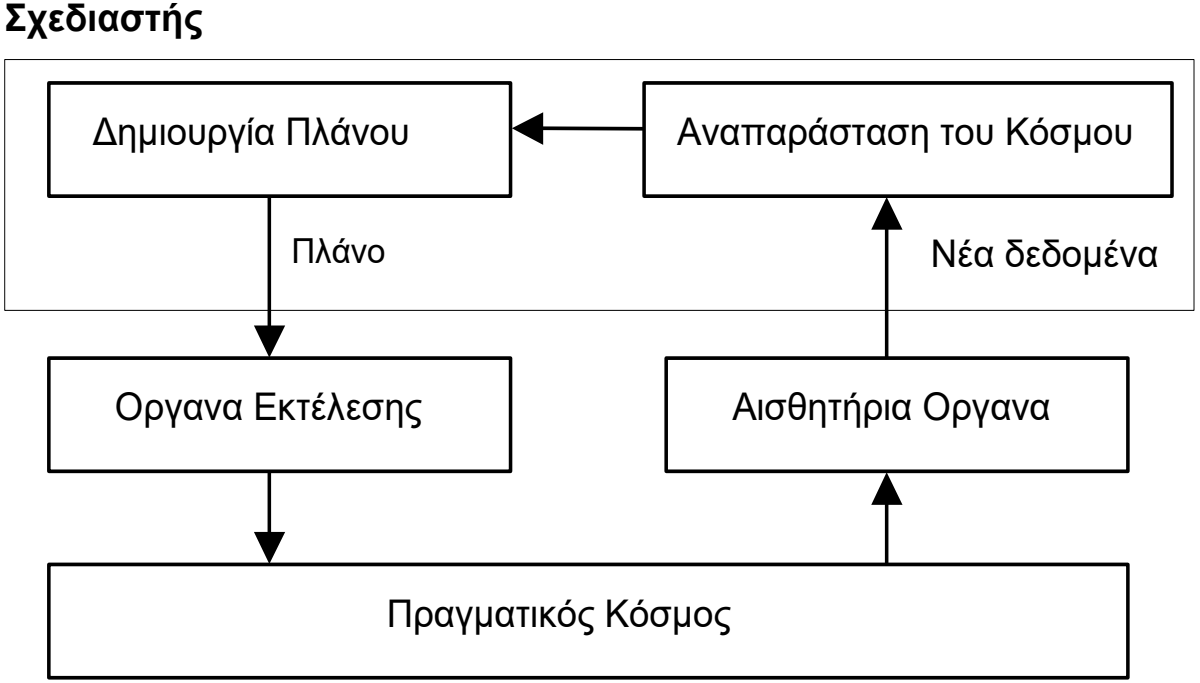
- Κατά την επιλογή μιας ενέργειας δεσμεύονται μόνο οι μεταβλητές αυτής που είναι απαραίτητες για την ενοποίηση ενός γεγονότος που η ενέργεια παράγει με την αντίστοιχη μη υποστηριζόμενη προϋπόθεση μιας άλλης ενέργειας.
- Επιτρέπονται δύο είδη περιορισμών δέσμευσης (binding constraints):
  - Οι περιορισμοί 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*).

Ερωτήσεις

- Το κριτήριο εφαρμοσιμότητας μιας ενέργειας *A* σε μια κατάσταση *S* σε σχέση με τις τρεις λίστες της *A* (*prec*(*A*), *add*(*A*), *del*(*A*)) δίνεται από τη σχέση:

  .....

- Η νέα κατάσταση *S'* που προκύπτει από την εφαρμογή μιας ενέργειας *A* σε μια κατάσταση *S* σε σχέση με τις τρεις λίστες της *A* (*prec(A)*, *add(A)*, *del(A)*) δίνεται από τον τύπο:

  .....

- Οι ενέργειες στο STRIPS αναπαριστώνται μέσω 3 λιστών:

  1).....
  2).....
  3).....

- Αναφέρετε 3 από τις παραδοχές του μοντέλου STRIPS.

  .....

- Προβλήματα σχεδιασμού ενεργειών (planning problems) είναι αυτά στα οποία ζητούμενο είναι.....

  .....

- Σχεδιαστής (planner) ή σύστημα σχεδιασμού (planning system) ονομάζεται το πρόγραμμα, το οποίο..... .....

- Τι είναι γραμμικό και τι μη-γραμμικό πλάνο;

Κυκλώστε το αντίστοιχο γράμμα Σ (ωστό) – Λ (άθος) στις επόμενες ερωτήσεις:		
Η τεχνική ανάλυσης των μέσων και των στόχων στον Σχεδιασμό Ενεργειών χρησιμοποιείται για την δημιουργία των πλάνων	Σ	Λ
Πλεονέκτημα του μοντέλου STRIPS είναι η δυνατότητα αποδοτικής περιγραφής της έννοιας του χρόνου	Σ	Λ
Το σύστημα STRIPS χρησιμοποιεί αναζήτηση στο χώρο των καταστάσεων	Σ	Λ
Η υπόθεση του κλειστού κόσμου αναφέρεται στην ιδιότητα του συστήματος σχεδιασμού να έχει πλήρη γνώση για την τρέχουσα κατάσταση του κόσμου	Σ	Λ
Το σύστημα σχεδιασμού STRIPS δεν είναι αποδοτικό για δυναμικούς κόσμους	Σ	Λ
Ένα πλεονέκτημα του STRIPS είναι η δυνατότητα δημιουργίας και καταστροφής νέων αντικειμένων κατά τη διάρκεια του σχεδιασμού.	Σ	Λ
Η λίστα διαγραφής στην αναπαράσταση STRIPS είναι υποσύνολο της λίστας προϋποθέσεων	Σ	Λ

Ασκήσεις

- 15.1 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.

15.1 Λύση

```
initial([αντικείμενο (obj1), ακατέργαστο(obj1), αγυάλιστο(obj1),
άβαφο(obj1)]).

goals([βαμένο(obj1)]).

operator(κατεργασία(X),
  [αντικείμενο(X), ακατέργαστο(X)],
  [ακατέργαστο(X)],
  [κατεργασμένο(X)]).

operator(γυάλισμα (X),
  [αντικείμενο(X), αγυάλιστο (X)],
  [αγυάλιστο(X)],
  [γυαλισμένο(X)]).

operator(βαφή (X),
  [αντικείμενο(X), άβαφο (X)],
  [άβαφο(X)],
  [βαμμένο(X)]).
```

Άσκηση 15.2

- Έστω ένα πρόβλημα σχεδιασμού που περιγράφεται από τα παρακάτω:
  - Αρχική κατάσταση $I = \{p,q,r,v\}$, Στόχοι $G = \{w,y,z,r,x,b,p\}$
  - 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 $h(S,G) = |G\text{-}S|$, 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)

15.2 Λύση

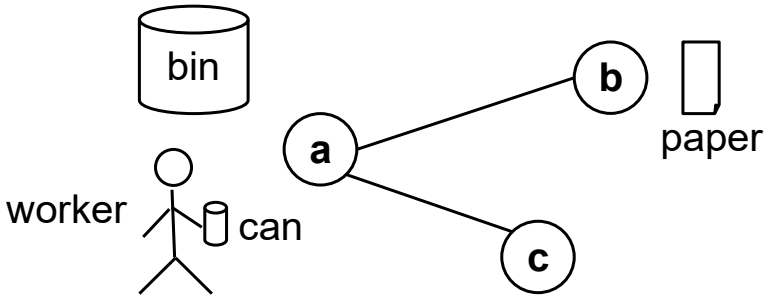
Βήμα	Κατάσταση	Ενέργειες	Επιλέγεται
1	$I=\{p,q,r,v\}$	A_1: {q,r,v,w}, h(A_1) = 7-2=5 A_4: {p,q,r,b,y,w}, h(A_4) = 7-5=2	A_4
2	{p,q,r,b,y,w}	A_3: {p,r,b,y,w,a,z}, h(A_1) = 7-6=1	A_3
3	{p,r,b,y,w,a,z}	A_2: {p,r,b,y,w,z,x,v}, h(A_1) = 7-7=0	A_2
4	{p,r,b,y,w,z,x,v}	$\supseteq$ G, επιστρέφεται το πλάνο <A_4,A_3,A_2>	

Ασκήσεις

- Έστω η κατάσταση S ={z, q, r} και οι ενέργειες:
  - ✓ ενέργεια α με προϋποθέσεις {q,z} και αποτελέσματα {+r, -q}
  - ✓ ενέργεια β με προϋποθέσεις {m,p} και αποτελέσματα {+z,-p}
  - ❑ Ελέγξτε αν οι ενέργειες α και β μπορούν να εφαρμοστούν στην S και δώστε τις καταστάσεις που προκύπτουν από την εφαρμογή τους.
- Έστω το πρόβλημα της μεταφοράς αυτοκινήτων με ένα οχηματαγωγό πλοίο. Στο πρόβλημα αυτό υπάρχουν κάποια λιμάνια, ένα οχηματαγωγό πλοίο και διάφορα αυτοκίνητα που βρίσκονται σε κάποιο λιμάνι και πρέπει να μεταφερθούν σε κάποιο άλλο.
  - ❑ Θεωρώντας ότι η μετακίνηση του οχηματαγωγού από ένα λιμάνι σε κάποιο άλλο γίνεται με μία κίνηση και ότι το οχηματαγωγό μπορεί να ταξιδεύει είτε άδειο, είτε μεταφέροντας αυτοκίνητα (υποθέστε ότι πάντα η χωρητικότητα του πλοίου είναι μεγαλύτερη από τα αυτοκίνητα)
  - ❑ αναπαραστήσετε σύμφωνα με το μοντέλο STRIPS το πρόβλημα.

Ασκήσεις

- Έστω το πρόβλημα του καθαρισμού μιας πόλης από απορρίμματα.
  - Στο πρόβλημα αυτό θεωρήστε ότι η πόλη αποτελείται από σημεία που ενώνονται μεταξύ τους μέσω ενός γράφου και υπάρχει ένας υπάλληλος καθαριότητας ο οποίος μετακινείται (με τα πόδια) από ένα σημείο Α σε ένα άλλο σημείο Β (αν βέβαια τα Α και Β ενώνονται μεταξύ τους).
  - Σε κάποια σημεία της πόλης υπάρχουν κάδοι απορριμμάτων στους οποίους ο υπάλληλος θα πρέπει να τοποθετήσει τα απορρίμματα που είναι διασκορπισμένα στην πόλη.
  - Η λογική που ακολουθεί ο υπάλληλος και η οποία απεικονίζεται στο σχήμα, είναι να περιφέρεται μέσα στην πόλη μαζεύοντας απορ-ρίμματα (ένα ή περισσότερα) και όταν βρεθεί σε κάποιο κάδο, να τα τοποθετεί σε αυτόν



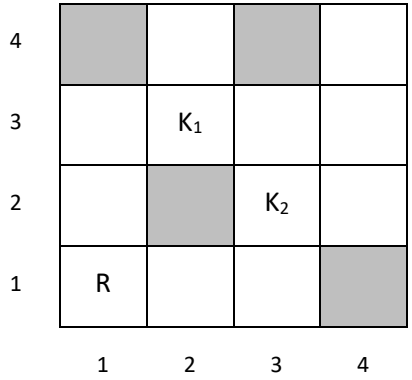
  - Η αρχική κατάσταση περιγράφεται ως:
    - ✓ START={point(a), point(b), point(c), connect(a,b), connect(a,c),
    - ✓ trashbin(bin), at(bin,a), trash(paper), at(paper,b), trash(can),
    - ✓ holding(can), at_worker(a) }
  - ενώ η τελική κατάσταση περιγράφεται με τους στόχους:
    - ✓ END={ in(paper,bin), in(can,bin) }
  - Να αναπαραστήσετε σύμφωνα με το μοντέλο STRIPS τους τελεστές του προβλήματος.

Ασκήσεις

- Μια βιοτεχνία κατασκευάζει παπούτσια από πρώτες ύλες.
  - Η κατασκευή ενός παπουτσιού περιλαμβάνει τρία στάδια:
    - ✓ επεξεργασία της σόλας, επεξεργασία του δέρματος και ράψιμο του παπουτσιού.
  - Η επεξεργασία της σόλας και η επεξεργασία του δέρματος μπορεί να γίνουν με οποιαδήποτε σειρά, όμως για να ραφτεί το παπούτσι πρέπει να έχουν ολοκληρωθεί οι δυο επιμέρους επεξεργασίες.
    - ✓ Να σημειωθεί ότι οι επεξεργασίες είναι διαφορετικές, ανάλογα με το αν πρόκειται για αριστερό ή δεξιό παπούτσι.
  - Να γραφούν τελεστές οι οποίοι να περιγράφουν τα παραπάνω βήματα κατασκευής ενός παπουτσιού.
  - Να οριστεί επίσης η αρχική και η τελική κατάσταση, αν θεωρηθεί ότι υπάρχουν 2 ακατέργαστα κομμάτια σόλας και δέρματος και πρέπει να κατασκευαστεί ένα ζευγάρι παπούτσια.

Ασκήσεις

- Ένα ρομπότ μπορεί να μετακινείται πάνω σε ένα πλέγμα θέσεων με διαστάσεις 4x4.
  - Στο πλέγμα υπάρχουν εμπόδια και κλειδιά.
  - Το ρομπότ δεν μπορεί να πάει στις θέσεις όπου υπάρχουν εμπόδια ενώ μπορεί να κρατά ένα ή κανένα κλειδί.
  - Για να πιάσει ένα κλειδί, το ρομπότ πρέπει να βρίσκεται στην ίδια θέση με το κλειδί.
  - Όταν αφήνει ένα κλειδί, αυτό τοποθετείται στη θέση όπου βρίσκεται το ρομπότ.
  - Στο σχήμα φαίνεται η αρχική κατάσταση του προβλήματος: με γκρι απεικονίζονται οι θέσεις των εμποδίων, R είναι το ρομπότ και K1 και K2 είναι δύο κλειδιά.
  - Στην τελική κατάσταση το κλειδί K1 πρέπει να είναι στη θέση (3, 1) ενώ το κλειδί K2 στη θέση (4, 4).
  - Να περιγραφεί η αρχική κατάσταση και η τελική κατάσταση.
  - Να γραφούν οι τελεστές του προβλήματος στην αναπαράσταση STRIPS.
  - Να κατασκευαστούν τα 7 πρώτα επίπεδα (0-6) του γράφου σχεδιασμού.
  - Από το γράφο σχεδιασμού να κατασκευαστεί η αναπαράσταση του προβλήματος με τη μορφή σύζευξης διαζεύξεων



Παράρτημα