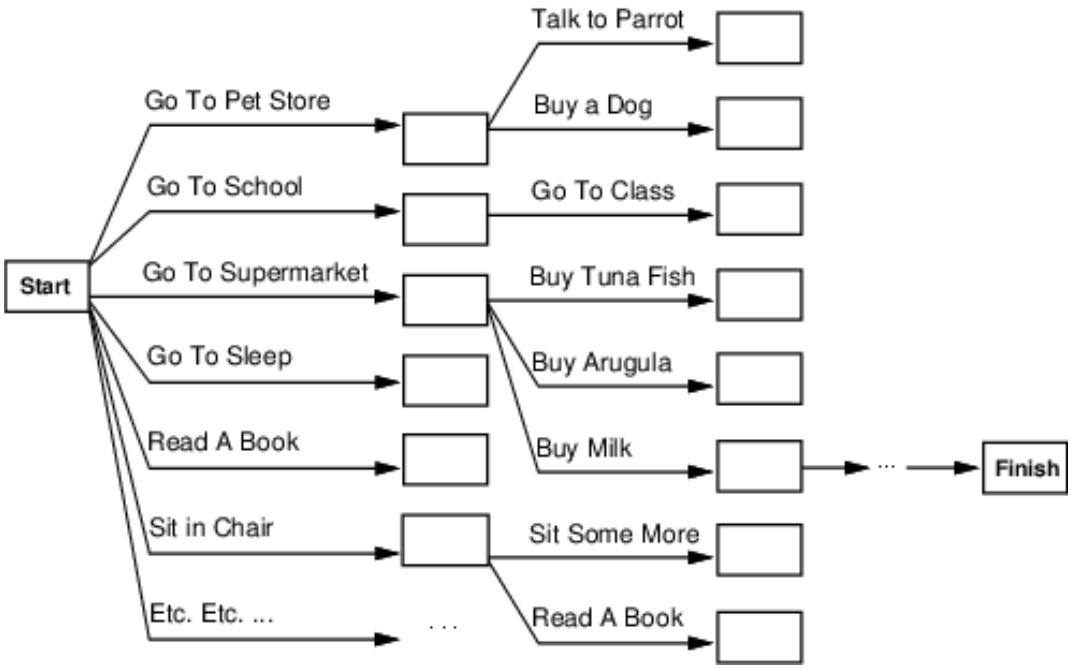
Τεχνητή Νοημοσύνη

Σχεδιασμός - Planning

Κάποια στοιχεία είναι δανεισμένα από τις διαφάνειες του P. Koehn (John Hopkins University), και του βιβλίου «Τεχνητή Νοημοσύνη» των Ι. Βλαχάβα κ.α.

Αναζήτηση vs. Σχεδιασμός

- Έστω το πρόβλημα του πραγματικού κόσμου: *Αγόρασε γάλα, μπανάνες και πριόνι.*
- Οι κλασσικοί αλγόριθμοι μάθησης αποτυγχάνουν
  - Πάρα πολλές οι καταστάσεις

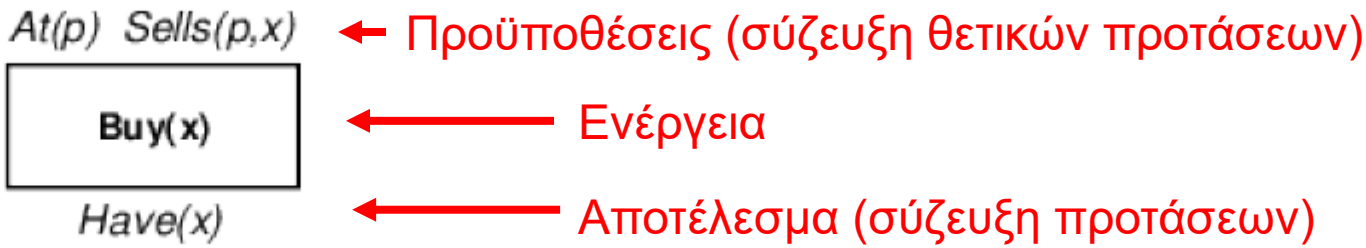


Αναζήτηση vs. Σχεδιασμός

- Στον Σχεδιασμό
  - η αναπαράσταση καταστάσεων είναι βελτιωμένη, με αποτέλεσμα να βοηθάει στην επιλογή
  - μειώνεται ο χώρος αναζήτησης με τη θέση ενδιάμεσων στόχων
  - η λύση δεν είναι απαραίτητα μια συνεχής ακολουθία μεταβάσεων από την αρχική κατάσταση, αλλά μια σειρά περιορισμών πάνω σε δράσεις

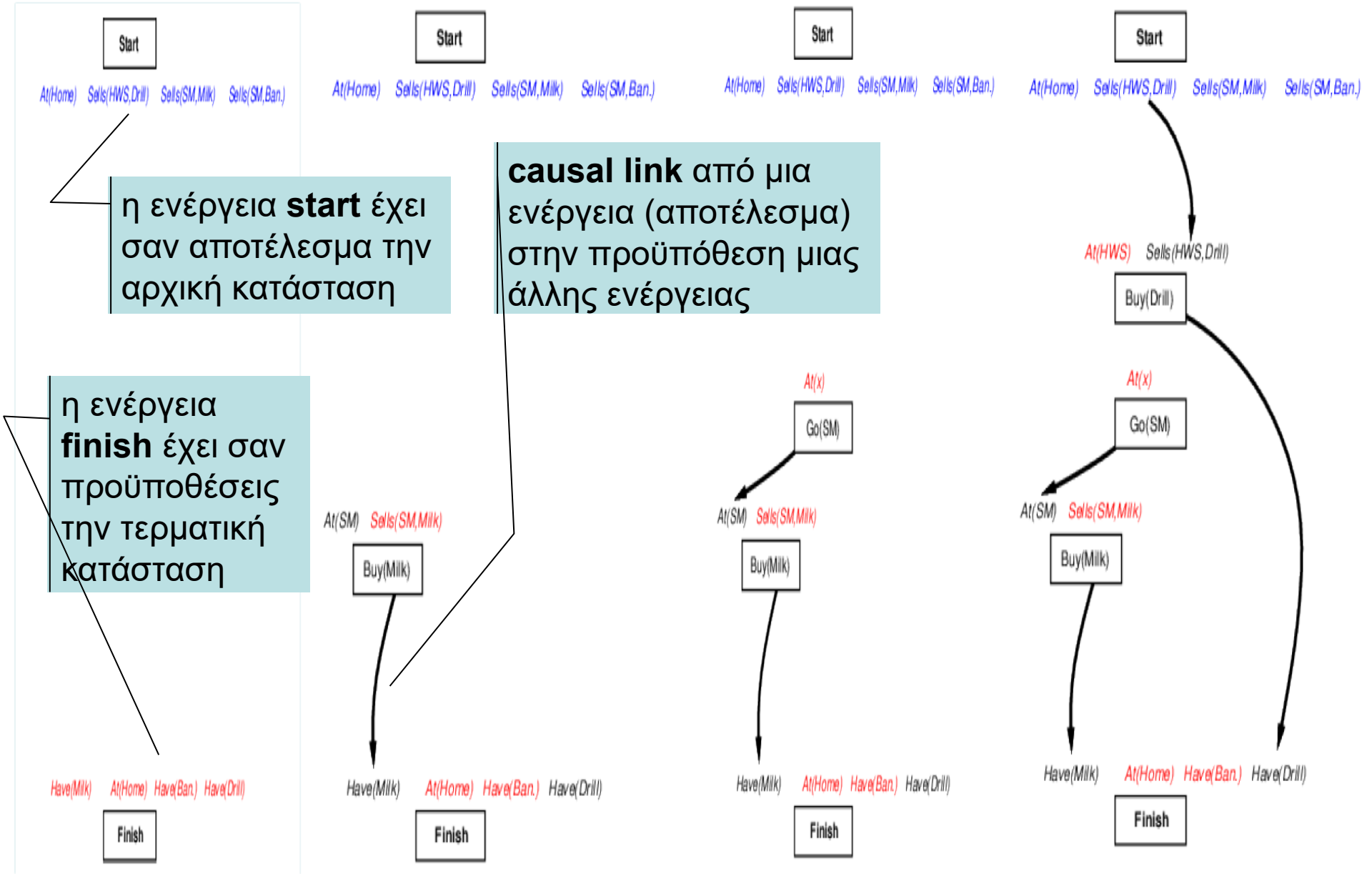
	Search	**Planning**
States	Data structures	Logical sentences
Actions	Program code	Preconditions/outcomes
Goal	Program code	Logical sentence (conjunction)
Plan	Sequence from S_0	Constraints on actions

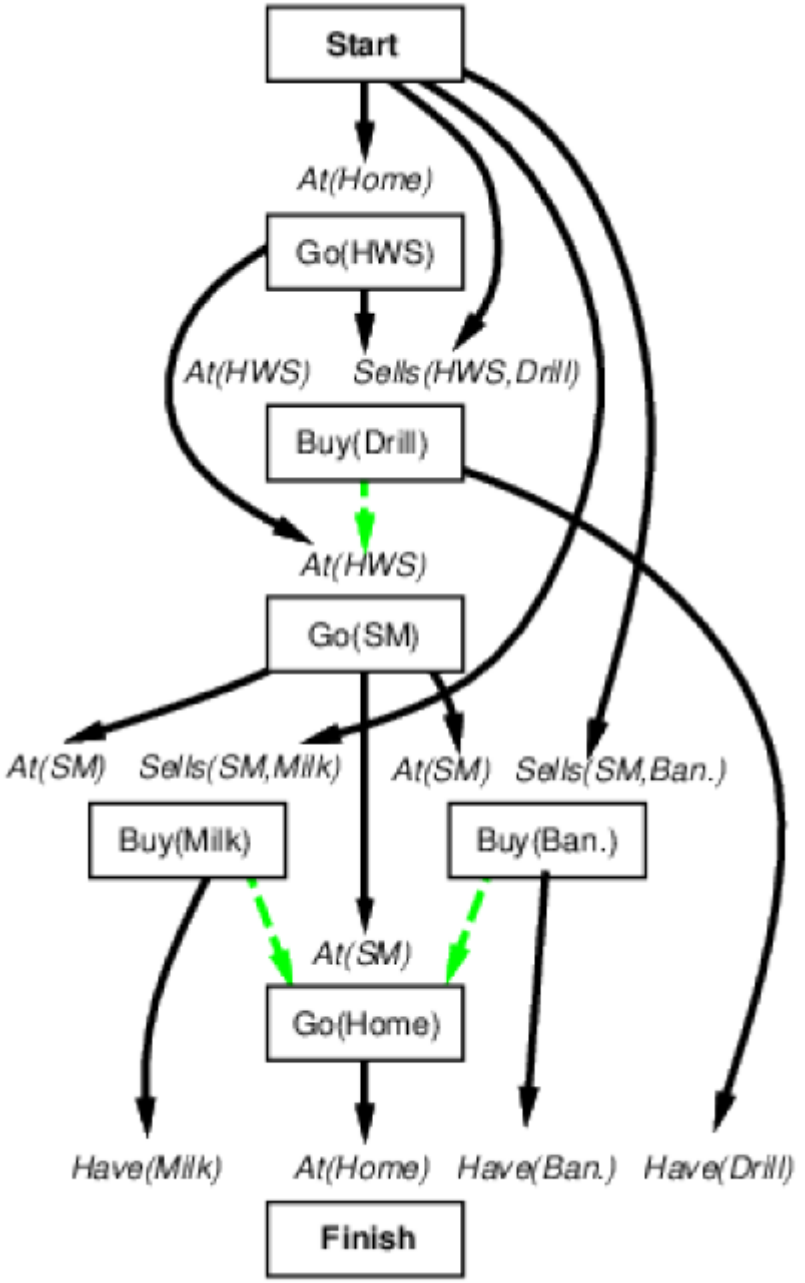
STanford Research Institute Problem Solver Operators (Τελεστές STRIPS)



ACTION: $Buy(x)$
PRECONDITION: $At(p), Sells(p, x)$
EFFECT: $Have(x)$

Παράδειγμα: Αγορά, Πώληση, Κατοχή





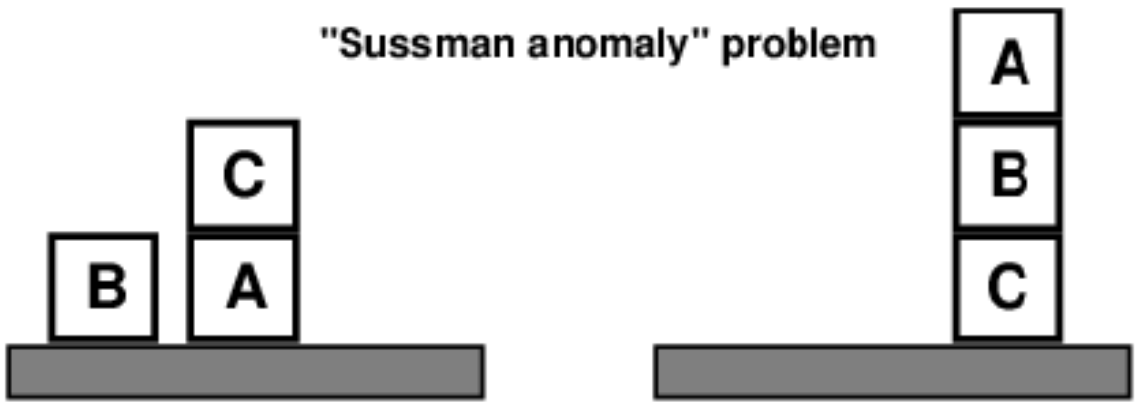
- Ένα σχέδιο είναι **ολοκληρωμένο** όταν έχει ικανοποιηθεί κάθε προϋπόθεση

- Μια προϋπόθεση έχει **ικανοποιηθεί** αν αποτελεί το αποτέλεσμα κάποιας προηγούμενης ενέργειας και δεν αναιρείται από κάποια άλλη ενδιάμεση ενέργεια

Διαδικασία Σχεδιασμού

- Εφαρμογή τελεστών σε μερικά σχέδια (partial plans)
  - Προσθήκη σύνδεσης (link) από μια υπάρχουσα ενέργεια σε μια προϋπόθεση που δεν έχει ικανοποιηθεί ακόμα (ανοιχτή προϋπόθεση - open condition)
  - Προσθήκη μιας ενέργειας για την ικανοποίηση μιας ανοιχτής προϋπόθεσης
  - Ταξινόμηση των ενεργειών ώστε να αποφευχθούν συγκρούσεις μεταξύ τους
- Σταδιακή μετάβαση από μη ολοκληρωμένα σχέδια σε ολοκληρωμένα ορθά σχέδια
- Οπισθοδρόμηση (Backtracking) αν
  - μια ανοιχτή προϋπόθεση δεν μπορεί να εκπληρωθεί
  - μια σύγκρουση δεν μπορεί να επιλυθεί

Παράδειγμα: Ο κόσμος των κύβων



Clear(x) On(x,z) Clear(y)

PutOn(x,y)

~On(x,z) ~Clear(y)
Clear(z) On(x,y)

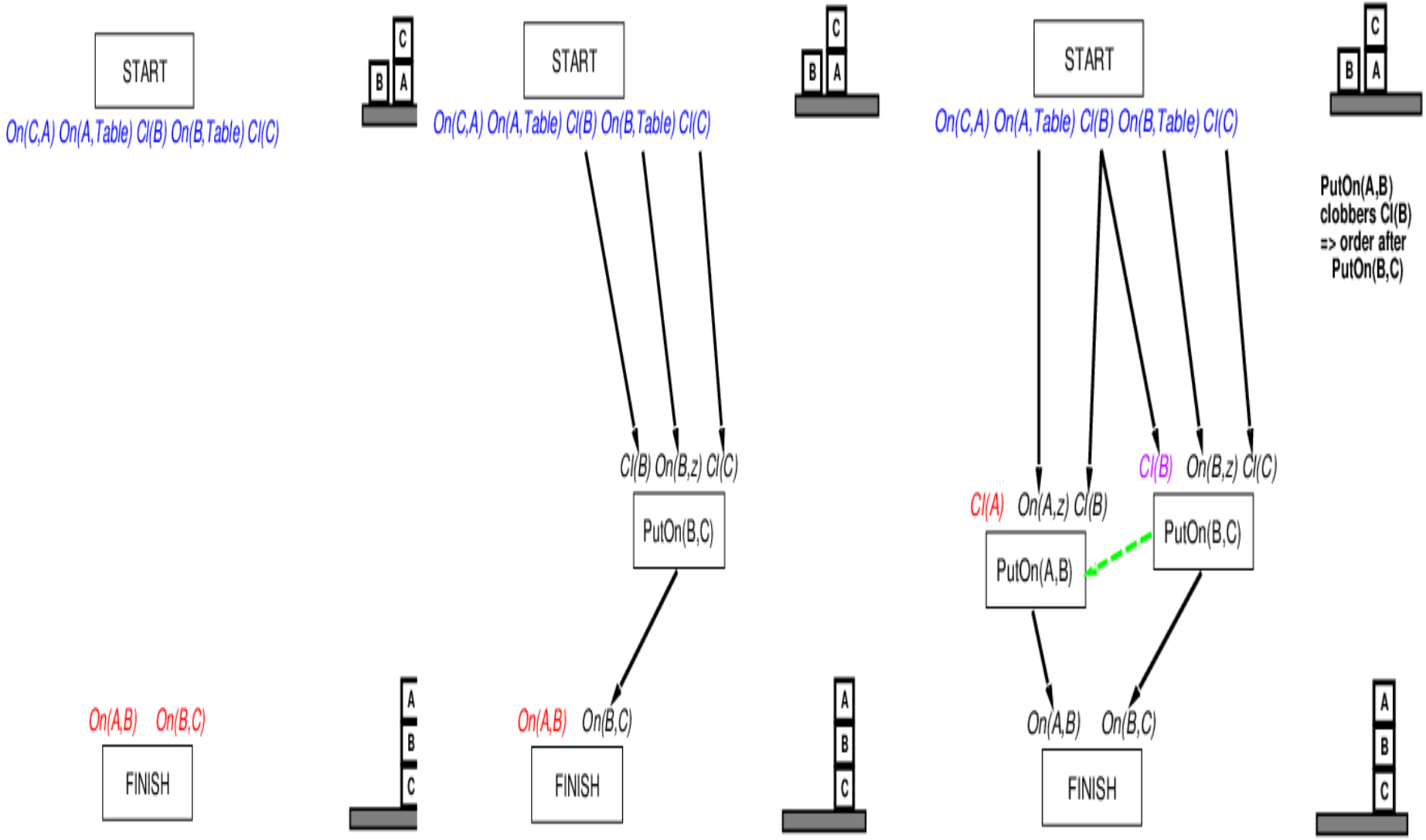
Clear(x) On(x,z)

PutOnTable(x)

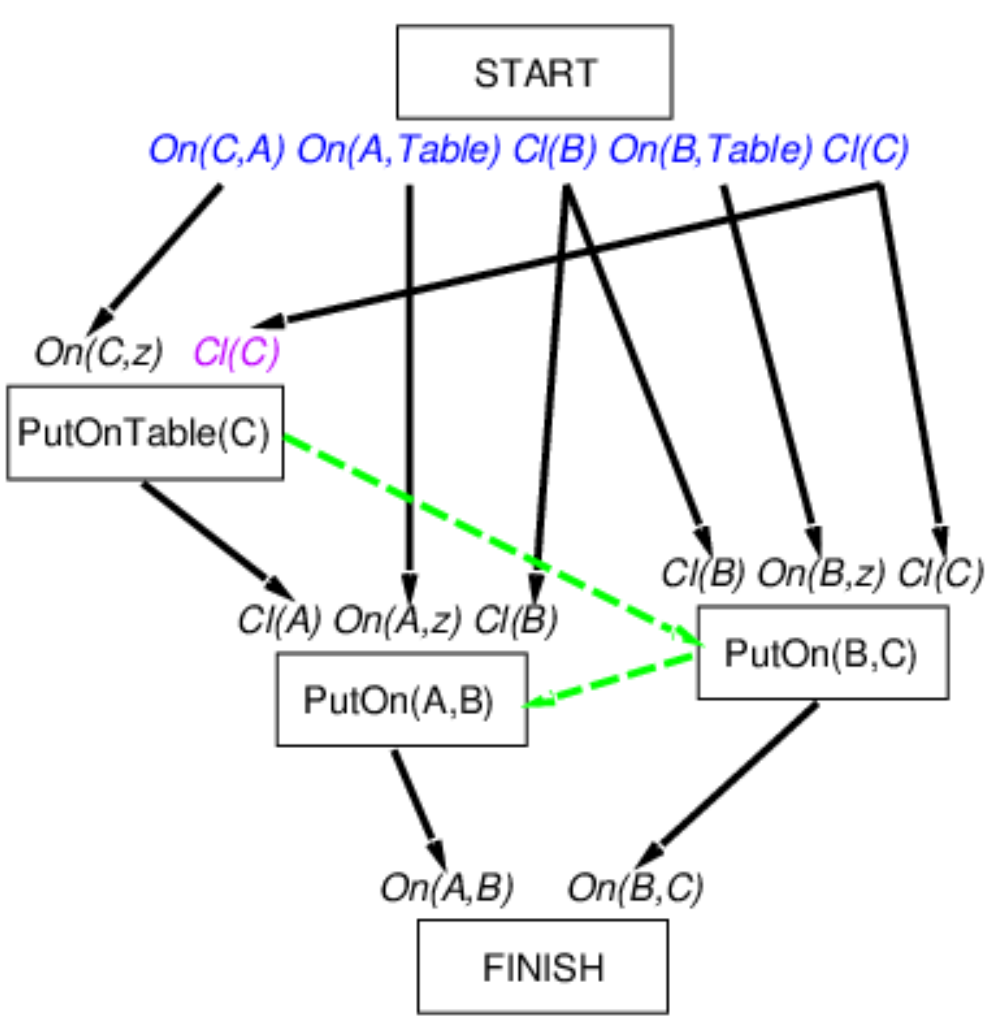
~On(x,z) Clear(z) On(x,Table)

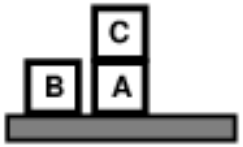
+ several inequality constraints

Παράδειγμα: Ο κόσμος των κύβων



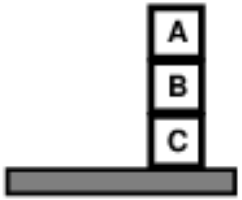
Παράδειγμα: Ο κόσμος των κύβων





PutOn(A,B)
clobbers Cl(B)
=> order after
PutOn(B,C)

PutOn(B,C)
clobbers Cl(C)
=> order after
PutOnTable(C)



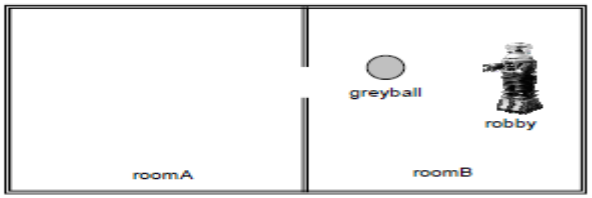
Παράδειγμα

Ένα πρόβλημα κίνησης και λαβής από ρομπότ

:

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

END = {at(greyball,roomA)}.



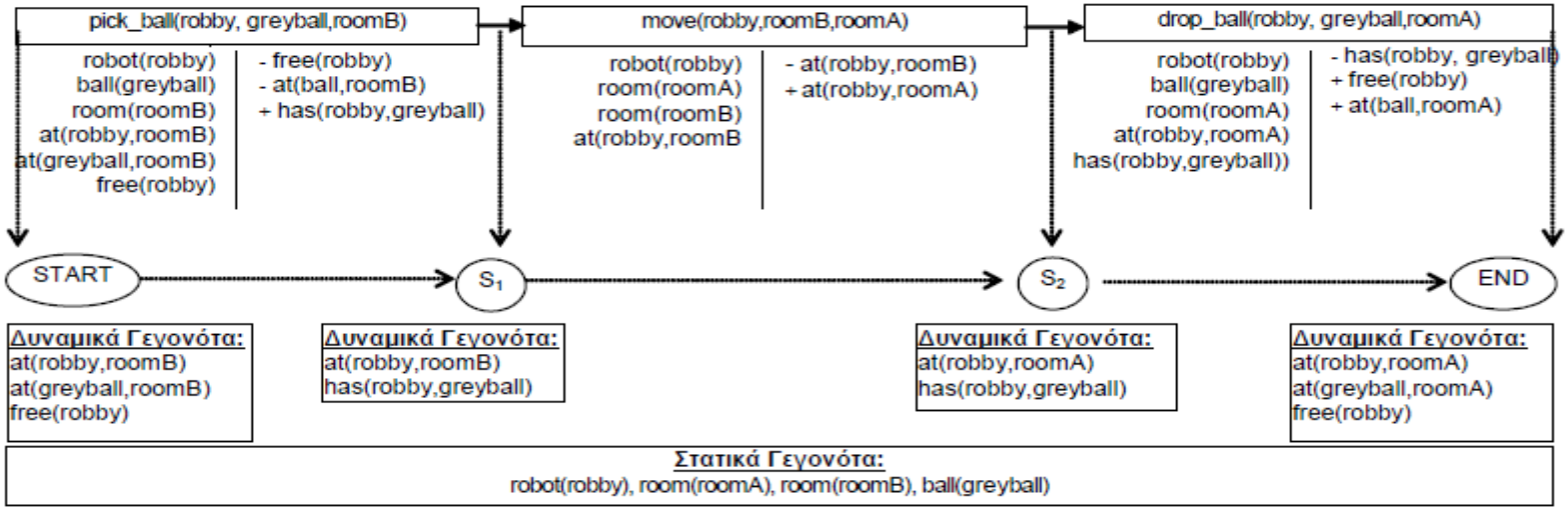
- Οι τελεστές είναι:

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

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)

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)

Ένα πλάνο είναι:



- Παράδειγμα: Πρόβλημα "Το Ρομπότ και το Κουτί"
- Φανταστείτε ένα ρομπότ σε ένα δωμάτιο με ένα κουτί. Ο στόχος είναι το κουτί να μεταφερθεί από το σημείο Α στο σημείο Β.

1. Αναπαράσταση Καταστάσεων (States):

Αρχική Κατάσταση (I): RobotAt(A), BoxAt(A), HandEmpty

Τελική Κατάσταση (G): BoxAt(B)

2. Τελεστές (Operators/Actions):

Οι τελεστές συνήθως ορίζονται με **Προϋποθέσεις (Preconditions)** και **Αποτελέσματα (Effects)**:

- **Operator: Go(x, y)** (Μετακίνηση ρομπότ από x σε y)
  - *Προϋποθέσεις:* RobotAt(x)
  - *Αποτελέσματα:* ¬RobotAt(x), RobotAt(y)
- **Operator: Pick(box, x)** (Σήκωμα κουτιού)
  - *Προϋποθέσεις:* RobotAt(x), BoxAt(x), HandEmpty
  - *Αποτελέσματα:* ¬HandEmpty, HandHolding(box), ¬BoxAt(x)
- **Operator: Drop(box, y)** (Απόθεση κουτιού)
  - *Προϋποθέσεις:* RobotAt(y), HandHolding(box)
  - *Αποτελέσματα:* HandEmpty, ¬HandHolding(box), BoxAt(y)

3. Πλάνο (Plan) - Η λύση:

Μια ακολουθία τελεστών που λύνει το πρόβλημα:

- Go(A, A) (ήδη εκεί)→Pick(Box, A)→Go(A, B)→Drop(Box, B)