**Τεχνητή Νοημοσύνη – 2η εργασία**

Ονοματεπώνυμο: Δεληγιαννάκης Χαράλαμπος

ΑΕΜ: 4383

**Πρώτο Ζητούμενο**

1. **Ανάλυση Πεδίου**

Οντότητες

* Προβλήτα (Dock)
* Κουτί (Container)
* Φορτηγό-Ρομπότ (Robot)

Σχέσεις

* Συνδέεται (isConnected) που συνδέει με γέφυρα δύο Προβλήτες μεταξύ τους.
* Έχει πιαστεί (isGrabbed) ένα Κουτί από Φορτηγό.
* Είναι ελεύθερο – βρίσκεται στην κορυφή (isOnTop) ένα Κουτί που δηλώνει ότι δεν υπάρχει άλλο Κουτί από πάνω του.
* Ανήκει σε στοίβα (isOnStack) ένα Κουτί που είναι στην κορυφή μιας στοίβας.

(Διευκρίνηση: ένα Κουτί που είναι στην κορυφή μιας στοίβας, θα είναι και isOnTop και isOnStack)

* Είναι απασχολημένο (isBusy) ένα Φορτηγό, δηλαδή κουβαλάει κάποιο Κουτί.
* Πάνω (on) που δηλώνει ότι ένα Κουτί είναι πάνω σε ένα άλλο Κουτί.
* Κουτί σε Προβλήτα (containerOnDock) που δηλώνει ότι ένα Κουτί προς το παρών βρίσκεται σε κάποια συγκεκριμένη Προβλήτα, και όχι σε Φορτηγό.
* Φορτηγό σε Προβλήτα (robotOnDock) που δηλώνει ότι ένα Φορτηγό προς το παρών βρίσκεται σε κάποια συγκεκρικμένη Προβλήτα.

Τελεστές

* **grabSingle(?r ?c ?d)**
* Προϋποθέσεις:

(robot ?r) (container ?c) (dock ?d) (not (isGrabbed ?c)) (isOnTop ?c) (not (isBusy ?r)) (not (isOnStack ?c)) (containerOnDock ?c ?d) (robotOnDock ?r ?d)

* Προσθέτει:

(isGrabbed ?c) (isBusy ?r)

* Αφαιρεί:

(isOnTop ?c) (containerOnDock ?c ?d)

* **grabFromStack(?r ?c ?cUnder ?d)**
* Προϋποθέσεις:

(robot ?r) (container ?c) (container ?cUnder) (dock ?d)

(not (isGrabbed ?c)) (isOnTop ?c) (isOnStack ?c) (not (isBusy ?r))

(on ?c ?cUnder) (containerOnDock ?c ?d) (robotOnDock ?r ?d)

* Προσθέτει:

(isGrabbed ?c) (isBusy ?r) (isOnTop ?cUnder)

* Αφαιρεί:

(isOnTop ?c) (on ?c ?cUnder) (containerOnDock ?c ?d)

(isOnStack ?c)

* **throwSingle(?r ?c ?d)**
* Προϋποθέσεις:

(robot ?r) (container ?c) (dock ?d) (isGrabbed ?c) (isBusy ?r) (robotOnDock ?r ?d)

* Προσθέτει:

(isOnTop ?c) (containerOnDock ?c ?d)

* Αφαιρεί:

(isBusy ?r)) (not (isGrabbed ?c)

* **throwToStack(?r ?c ?cUnder ?d)**
* Προϋποθέσεις:

(robot ?r) (container ?c) (dock ?d) (container ?cUnder) (isGrabbed ?c) (isBusy ?r) (isOnTop ?cUnder) (robotOnDock ?r ?d)

(containerOnDock ?cUnder ?d)

* Προσθέτει:

(isOnTop ?c) (containerOnDock ?c ?d) (on ?c ?cUnder) (isOnStack ?c)

* Αφαιρεί:

(isBusy ?r) (isGrabbed ?c) (isOnTop ?cUnder)

* **robotMove(?r ?d1 ?d2)**
* Προϋποθέσεις:

(robot ?r) (dock ?d1) (dock ?d2) (robotOnDock ?r ?d1)

(isConnected ?d1 ?d2)

* Προσθέτει:

(robotOnDock ?r ?d2)

* Αφαιρεί:

(robotOnDock ?r ?d1)

1. **Ανάλυση Προβλήματος**

Έστω πρόβλημα στο οποίο υπάρχουν 1 Φορτηγό, 3 Κουτιά και 3 Προβλήτες. Δηλαδή έχει ως αντικέιμενα:

* r1, c1, c2, c3, d1, d2, d3

Με αρχική κατάσταση:

* (robot r1)

(container c1) (container c2) (container c3)

(dock d1) (dock d2) (dock d3)

(isConnected d1 d3) (isConnected d1 d2) (isConnected d3 d1) (isConnected d2 d1)

(containerOnDock c1 d1) (containerOnDock c3 d1)

(containerOnDock c2 d2)

(robotOnDock r1 d2)

(isOnTop c2) (isOnTop c3)

(isOnStack c3)

(on c3 c1)

Και με τελική κατάσταση – στόχο:

* (containerOnDock c1 d3) (containerOnDock c3 d3) (containerOnDock c2 d3)

(robotOnDock r1 d1)

(isOnTop c1) (isOnTop c3)

(isOnStack c1)

(on c1 c2)

Δηλαδή, οπτικά είναι το εξής:

A black arrow pointing to the right

Description automatically generated

**Δεύτερο Ζητούμενο**

Γράφτηκε κώδικας σε γλώσσα PDDL για επιλύει ειδικά το παραπάνω πρόβλημα που περιγράφτηκε. Η γρηγορότερη αποδοτική λύση δίνεται όταν ο κώδικας τρέχει με τον αλγόριθμο **top-k planner** και το γρηγορότερο πλάνο είναι το 2ο, οπότε ως k πρέπει να δωθεί ένας αριθμός μεγαλύτερος ή ίσος του 2. Ο αριθμός που παράγει το συγκεκριμένο πλάνο είναι 13, δηλαδή όσος χρειάζεται για να επιτευχθεί η γρηγορότερη αποδοτική λύση.

Ο κώδικας υπάρχει στα αρχεία *domain.pddl* και *prob.pddl*