ΤΕΧΝΗΤΗ ΝΟΗΜΟΣΥΝΗ

Το πρόβλημα του Parking

ΠΑΝΤΕΛΕΗΜΩΝ ΠΡΩΙΟΣ ΔΕΣΠΟΙΝΑ ΛΥΚΟΥΔΗ

ice18390023 ice18390151

5ο Εξάμηνο 5ο Εξάμηνο

[ice18390023@uniwa.gr](mailto:ice18390023@uniwa.gr) [ice18390151@uniwa.gr](mailto:ice18390151@uniwa.gr)

Τμήμα 1 (Νέοι) Δευτέρα 17:00-19:00



ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

UNIVERSITY OF WEST ATTICA

Υπεύθυνοι Καθηγητές

ΓΕΩΡΓΟΥΛΗ ΑΙΚΑΤΕΡΙΚΗ

ΒΟΥΛΟΔΗΜΟΣ ΑΘΑΝΑΣΙΟΣ

ΚΟΛΛΙΑ ΗΛΙΑΝΝΑ

ΤΣΕΛΕΝΤΗ ΠΑΝΑΓΙΩΤΑ

Τμήμα Μηχανικών Πληροφορικής και Υπολογιστών

2020-2021

Περιεχόμενα

[1. Ο κόσμος του προβλήματος (απλή εκδοχή) 3](#_Toc61903142)

[1.1 Το πρόβλημα του parking 3](#_Toc61903143)

[1.2 Αρχική Κατάσταση 3](#_Toc61903144)

[1.3 Τελική Κατάσταση 3](#_Toc61903145)

[2. Αναπαράσταση του κόσμου του προβλήματος (απλή εκδοχή) 4](#_Toc61903146)

[2.1 Γεγονότα του προβλήματος 4](#_Toc61903147)

[2.2 Αρχικοποίηση Γεγονότων 5](#_Toc61903148)

[2.3 Σχολιασμός των κανόνων 5](#_Toc61903149)

[2.3.1 swapNorth 5](#_Toc61903150)

[2.3.2 swapEast 6](#_Toc61903151)

[2.3.3 carEnters 6](#_Toc61903152)

[2.4 Στρατηγική 7](#_Toc61903153)

[3. Ο κόσμος του προβλήματος (ολοκληρωμένη εκδοχή) 7](#_Toc61903154)

[3.1 Το πρόβλημα του Parking 7](#_Toc61903155)

[3.2 Αρχική κατάσταση 8](#_Toc61903156)

[3.3 Τελική Κατάσταση 8](#_Toc61903157)

[4. Αναπαράσταση του κόσμου του προβλήματος (ολοκληρωμένη εκδοχή) 8](#_Toc61903158)

[4.1 Επιπρόσθετα γεγονότα του προβλήματος 8](#_Toc61903159)

[4.2 Αρχικοποίηση γεγονότων 9](#_Toc61903160)

[4.3 Σχολιασμός των επιπρόσθετων κανόνων και αλλαγές που έγιναν 10](#_Toc61903161)

[4.3.1 changeLevel 10](#_Toc61903162)

[4.3.2 swapNorth 11](#_Toc61903163)

[4.3.3 swapEast 11](#_Toc61903164)

[4.3.4 swapLevelE 12](#_Toc61903165)

[4.3.5 swapLevelF 12](#_Toc61903166)

[4.3.6 carEnters 12](#_Toc61903167)

[4.3.7 goalFound 13](#_Toc61903168)

[4.3.8 howManyCarsWaiting 13](#_Toc61903169)

[4.3.9 registerCar 13](#_Toc61903170)

# 1. Ο κόσμος του προβλήματος (απλή εκδοχή)

## 1.1 Το πρόβλημα του parking

Στο πρόβλημα της εκφώνησης, υπάρχει ένα parking που λειτουργεί με αυτόματο παρκαδόρο για την είσοδο στο χώρο και την τοποθέτηση σε πλατφόρμες παρκαρίσματος των αυτοκινήτων που είναι σε αναμονή.

Ένας χώρος στάθμευσης (parking), αρχικά διαθέτει ένα επίπεδο, που περιέχει 6 χώρους (spaces) υποδοχής πλατφορμών, με χώρο εισόδου τον S02. Κάθε χώρος (space) χαρακτηρίζεται από το όνομα του, από το επίπεδο στο οποίο βρίσκεται και από την κατάσταση του, δηλαδή αν είναι ελεύθερος ή αν υπάρχει επάνω τοποθετημένη πλατφόρμα.

Επιπλέον, υπάρχουν 5 πλατφόρμες (platforms), αριθμημένες από P1 έως P5. Κάθε πλατφόρμα (platform) χαρακτηρίζεται από το όνομα της , την κατάσταση της (άδεια ή με παρκαρισμένο αυτοκίνητο) και τον χώρο πάνω στον οποίο βρίσκεται. Στην αρχή του λειτουργίας του parking, όλες οι πλατφόρμες είναι άδειες και τοποθετημένες στους χώρους των τριών επιπέδων και στο χώρο υποδοχής δεν υπάρχει πλατφόρμα. Μια πλατφόρμα μπορεί να κινηθεί σε ένα γειτονικό της χώρο του ίδιου επιπέδου αρκεί αυτός να είναι κενός.

Για να μπορέσει ένα αυτοκίνητο (car) να μπει στο parking, θα πρέπει να υπάρχει τουλάχιστον μια ελεύθερη πλατφόρμα σε οποιονδήποτε χώρο του επιπέδου του parking και να μετακινηθεί από τον αυτόματο παρκαδόρο στον χώρο εισόδου. Επίσης, ο οδηγός αφήνοντας το αμάξι του στο χώρο αναμονής, θα πρέπει να το δηλώσει στο σύστημα παρακολούθησης, δίνοντας τον αριθμό κυκλοφορίας του.

## 1.2 Αρχική Κατάσταση

Στην αρχική κατάσταση του προβλήματος, υπάρχουν 6 αυτοκίνητα σε αναμονή και οι πλατφόρμες είναι τοποθετημένες στους υπάρχοντες χώρους όπως στο Σχήμα 1.1.

|  |  |  |
| --- | --- | --- |
| Space 4  Platform 3 EMPTY | Space 5  Platform 4 EMPTY | Space 6  Platform 5 EMPTY |
| Space 1  Platform 1 EMPTY | Space 2  EMPTY EMPTY | Space 3  Platform 2 EMPTY |

Σχήμα 1.1: Κάτοψη επιπέδου 1

## 1.3 Τελική Κατάσταση

Ως τελική κατάσταση – στόχο της συγκεκριμένης εκδοχής του προβλήματος θα πρέπει να υπάρχει μόνο ένα αμάξι σε κατάσταση waiting, ενώ όλα τα υπόλοιπα αμάξια να έχουν τοποθετηθεί στο parking.

|  |  |  |
| --- | --- | --- |
| Space 4  Platform 5 FULL | Space 5  Platform 2 FULL | Space 6  Platform 1  FULL |
| Space 1  EMPTY EMPTY | Space 2  Platform 4  FULL | Space 3  Platform 2 FULL |

1 Car is waiting

Σχήμα 1.2: Τελική Κατάσταση

# 2. Αναπαράσταση του κόσμου του προβλήματος (απλή εκδοχή)

## 2.1 Γεγονότα του προβλήματος

Για την αναπαράσταση του προβλήματος του Parking, χρειαστήκαμε ένα πρότυπο SPACE, το οποίο αναφέρεται στους χώρους του Parking και έχει τρεις ιδιότητες όνομα, επίπεδο, κατάσταση (με πλατφόρμα ή χωρίς).

(deftemplate space

(slot name)

(slot level)

(slot status)

)

Στην συνέχεια χρειαστήκαμε ένα πρότυπο PLATFORM, το οποίο περιγράφει τα δεδομένα για τις πλατφόρμες, δηλαδή το όνομα τους, το χώρο (στον οποίο βρίσκονται) και την κατάσταση του (κενή ή με αμάξι)

(deftemplate platform

(slot name)

(slot space)

(slot status)

)

Επιπλέον, δημιουργήσαμε ένα template με όνομα CAR, το οποίο δίνει τις εξής ιδιότητες των αμαξιών αριθμό (πινακίδας) και κατάσταση (εάν περιμένει ή εάν είναι ήδη στο Parking).

(deftemplate car

(slot plate)

(slot status)

)

Επιπρόσθετα, χρειαστήκαμε και δύο templates, τα οποία θα περιέχουν ιδιότητες για τη γειτνίαση των χώρων στο επίπεδο, δηλαδή σε ποιόν χώρο βρισκόμαστε και ποιος είναι ο γειτονικός. Το πρώτο template ονομάστηκε East και στη δεύτερη ιδιότητα περιέχει πληροφορίες για το γειτονικό χώρο στα δεξιά, ενώ το δεύτερο template ονομάστηκε North και στη δεύτερη ιδιότητα περιέχει πληροφορίες για το γειτονικό χώρο από επάνω. Αυτές οι πληροφορίες μπορούν να χρησιμοποιηθούν και για το αντίστροφο γείτονα (αριστερά και κάτω).

(deftemplate east

(slot space)

(slot right\_space)

)

(deftemplate north

(slot space)

(slot up\_space)

)

Τέλος, χρησιμοποιήσαμε ένα template με όνομα parking, το οποίο περιέχει το όνομα του parking (δεν μας χρειάζεται), τον αριθμό αμαξιών σε αναμονή και τον αριθμό των ελεύθερων πλατφορμών.

(deftemplate parking

(slot name)

(slot carsWaiting (type INTEGER))

(slot emptyPlatforms (type INTEGER))

)

## 2.2 Αρχικοποίηση Γεγονότων

Από κάτω φαίνεται η αρχικοποίηση των γεγονότων της πρώτης εκδοχής του προβλήματος μας. Όπως προαναφέρθηκε στην αναμονή υπάρχει ένα αμάξι περισσότερο από αυτά που μπορούν να εισέλθουν στο parking.

(deffacts init

(space (name S01) (level 0) (status FULL))

(space (name S02) (level 0) (status EMPTY))

(space (name S03) (level 0) (status FULL))

(space (name S04) (level 0) (status FULL))

(space (name S05) (level 0) (status FULL))

(space (name S06) (level 0) (status FULL))

(platform (name P1) (space S01) (status EMPTY))

(platform (name P2) (space S03) (status EMPTY))

(platform (name P3) (space S04) (status EMPTY))

(platform (name P4) (space S05) (status EMPTY))

(platform (name P5) (space S06) (status EMPTY))

(car (plate ZHN3466) (status WAITING))

(car (plate ZZM4556) (status WAITING))

(car (plate IKM7845) (status WAITING))

(car (plate IEN3387) (status WAITING))

(car (plate IXK6876) (status WAITING))

(car (plate IKK2568) (status WAITING))

(parking (name myParking) (carsWaiting 6) (emptyPlatforms 5))

(east (space S01) (right\_space S02))

(east (space S02) (right\_space S03))

(east (space S04) (right\_space S05))

(east (space S05) (right\_space S06))

(north (space S01) (up\_space S04))

(north (space S02) (up\_space S05))

(north (space S03) (up\_space S06))

)

## 2.3 Σχολιασμός των κανόνων

### 2.3.1 swapNorth

Ο κανόνας swapNorth, ισχύει για κάποιο space, το οποίο είναι γεμάτο, έχει κάποιο γείτονα στο north που είναι ο άδειος χώρος και υπάρχει τουλάχιστον μια άδεια πλατφόρμα και ένα αυτοκίνητο σε αναμονή. Τότε, αλλάζει τα 2 γεγονότα για τους χώρους, όπου FULL σε EMPTY και το ανάποδο. Επίσης, αλλάζει το όνομα του χώρου της πλατφόρμας, σε αυτόν που μεταφέρθηκε.

(defrule swapNorth

?fSpace <- (space (name ?sp) (status FULL))

(or (north (space ?sp) (up\_space ?neigb))

(north (up\_space ?sp) (space ?neigb)))

?eSpace <- (space (name ?neigb) (status EMPTY))

?plat <- (platform (space ?sp))

(parking (carsWaiting ?cw) (emptyPlatforms ?ep))

(test (> ?cw 0))

(test (> ?ep 0))

=>

(modify ?fSpace (status EMPTY))

(modify ?eSpace (status FULL))

(modify ?plat (space ?neigb))

)

### 2.3.2 swapEast

Αντίστοιχα με την swapNorth, έτσι και η swapEast ισχύει για κάποιο space το οποίο είναι γεμάτο, έχει κάποιο γείτονα east, που είναι άδειος χώρος. Τότε, αλλάζει τα δύο γεγονότα για τους χώρους όπου FULL σε EMPTY και το ανάποδο και τέλος αλλάζει το όνομα του χώρου της πλατφόρμας σε αυτόν.

(defrule swapEast

?fSpace <- (space (name ?sp) (status FULL))

(or (east (space ?sp) (right\_space ?neigb))

(east (right\_space ?sp) (space ?neigb)))

?eSpace <- (space (name ?neigb) (status EMPTY))

?plat <- (platform (space ?sp))

(parking (carsWaiting ?cw) (emptyPlatforms ?ep))

(test (> ?cw 0))

(test (> ?ep 0))

=>

(modify ?fSpace (status EMPTY))

(modify ?eSpace (status FULL))

(modify ?plat (space ?neigb))

)

### 2.3.3 carEnters

Ο κανόνας carEnters, έχει προτεραιότητα 1 και ισχύει αν υπάρχουν αυτοκίνητα σε αναμονή και η υπάρχει πλατφόρμα στον χώρο 2, η οποία είναι ελεύθερη. Τότε, θα αλλάξει η πλατφόρμα σε γεμάτη, το αυτοκίνητο από WAITING θα πάρει το όνομα της πλατφόρμας και τέλος, θα μειώσουμε κατά 1 τις τιμές του parking.

(defrule carEnters

(declare (salience 1))

?plat <- (platform (name ?platN) (space S02) (status EMPTY))

?car <- (car (status WAITING))

?park <- (parking (carsWaiting ?cw) (emptyPlatforms ?ep))

(test (> ?cw 0))

(test (> ?ep 0))

=>

(modify ?plat (status FULL))

(modify ?car (status ?platN))

(modify ?park (carsWaiting (- ?cw 1)) (emptyPlatforms (- ?ep 1)))

)

## 2.4 Στρατηγική

Για την ολοκλήρωση του πρώτου μέρους του προβλήματος χρειάστηκε να χρησιμοποιήσουμε την στρατηγική breadth. Αυτό το κάναμε διότι με την στρατηγική depth έμπαινε μόνιμα σε μία επανάληψη που δεν τελείωνε ποτέ. Ο κανόνας υλοποίησης της στρατηγικής αναγράφεται από κάτω.

(defrule initStrategy

(declare (salience 10000))

(initial-fact)

=>

(set-strategy breadth)

)

# 3. Ο κόσμος του προβλήματος (ολοκληρωμένη εκδοχή)

## 3.1 Το πρόβλημα του Parking

Στην ολοκληρωμένη εκδοχή του προβλήματος του parking, ο χώρος στάθμευσης (parking) διαθέτει ένα πολυεπίπεδο με 3 επίπεδα (1-3) που το κάθε ένα επίπεδο περιέχει 6 χώρους (spaces) υποδοχής πλατφορμών. Κάθε χώρος (space) χαρακτηρίζεται από το όνομα του, από το επίπεδο στο οποίο βρίσκεται και από την κατάσταση του, δηλαδή αν είναι ελεύθερος ή αν υπάρχει επάνω τοποθετημένη πλατφόρμα.

Επιπλέον, υπάρχουν 14 πλατφόρμες (platforms), αριθμημένες από P1 έως P14. Κάθε πλατφόρμα (platform) χαρακτηρίζεται από το όνομα της , την κατάσταση της (άδεια ή με παρκαρισμένο αυτοκίνητο) και τον χώρο πάνω στον οποίο βρίσκεται. Στην αρχή του λειτουργίας του parking, όλες οι πλατφόρμες είναι άδειες και τοποθετημένες στους χώρους των τριών επιπέδων και στο χώρο υποδοχής δεν υπάρχει πλατφόρμα.

Μια πλατφόρμα μπορεί να κινηθεί σε ένα γειτονικό της χώρο του ίδιου επιπέδου αρκεί αυτός να είναι κενός. Για την δυνατότητα κίνησης μιας πλατφόρμας από όροφο σε όροφο, οι εμπρός κεντρικές θέσεις των ορόφων (πάνω από τον χώρο εισόδου) συγκροτούν ένα φρεάτιο στο οποίο είναι προσαρμοσμένα συρματόσκοινα που διευκολύνουν την κάθετη κίνηση μιας πλατφόρμας (προς τα πάνω ή προς τα κάτω).

Για να μπορέσει ένα αυτοκίνητο (car) να μπει στο parking, θα πρέπει να υπάρχει τουλάχιστον μια ελεύθερη πλατφόρμα σε οποιονδήποτε χώρο οποιουδήποτε επιπέδου του parking και να μετακινηθεί από τον αυτόματο παρκαδόρο στον χώρο εισόδου.

Όταν ένας οδηγός επιθυμεί να παρκάρει το αυτοκίνητό του, δίνει τα στοιχεία του στο αυτόματο παρκαδόρο που φροντίζει να εντοπίσει και να φέρει μια ελεύθερη πλατφόρμα στο χώρο εισόδου. Αντίστοιχα, όταν ένας οδηγός θέλει να παραλάβει το παρκαρισμένο αυτοκίνητό του, δίνει τα στοιχεία του αυτοκινήτου, ο αυτόματος παρκαδόρος εντοπίζει την πλατφόρμα πάνω στην οποία είναι τοποθετημένο το αυτοκίνητο, και αυτή μεταφέρεται αυτόματα και τοποθετείται στο χώρο εισόδου ώστε να παραλάβει ο οδηγός το αυτοκίνητο και να ελευθερώσει την πλατφόρμα

## 3.2 Αρχική κατάσταση

Κατά την αρχική κατάσταση, όλες οι πλατφόρμες του parking είναι άδειες και υπάρχουν ορισμένα αμάξια που περιμένουν. Η κάτοψη του επιπέδου 1 αναπαριστάται στο Σχήμα 1.1, ενώ των επιπέδων 2 και 3, απεικονίζονται στο Σχήμα 3.1 και 3.2 αντίστοιχα.

|  |  |  |
| --- | --- | --- |
| Space 10  Platform 8  Empty | Space 11  Platform 9  EMPTY | Space 12  EMPTY EMPTY |
| Space 7  EMPTY  EMPTY | Space 8  Platform 6  EMPTY | Space 9  Platform 7  EMPTY |

Σχήμα 3.1: Κάτοψη επιπέδου 2.

|  |  |  |
| --- | --- | --- |
| Space 16  Platform 12  EMPTY | Space 17  Platform 13  EMPTY | Space 18  Platform 14  EMPTY |
| Space 13  EMPTY  EMPTY | Space 14  Platform 10  EMPTY | Space 15  Platform 11  EMPTY |

Σχήμα 3.2: Κάτοψη επίπεδου 3.

## 3.3 Τελική Κατάσταση

Η κατάσταση στόχος είναι να μην υπάρχουν αυτοκίνητα στην αναμονή. Στην ενδιάμεση παράδωση δεν υπήρχε τέτοιος, έτσι ώστε να τερματίζει, αλλά την είχαν η swapEast και swapNorth στους ισχυρισμούς της, μαζί με τον αν υπάρχουν διαθέσιμες πλατφόρμες, έτσι ώστε να μην γίνονται άσκοπες μετακινήσεις.

# 4. Αναπαράσταση του κόσμου του προβλήματος (ολοκληρωμένη εκδοχή)

## 4.1 Επιπρόσθετα γεγονότα του προβλήματος

Γείτονας κάθε επιπέδου από τον current στον επάνω του.

(deftemplate level

(slot space)

(slot above\_space)

)

Ο αριθμός του επιπέδου όπου ασχολούμαστε αυτήν την στιγμή να γεμίσουμε έτσι ώστε να πάμε στον επόμενο

(deftemplate currentLevel

(slot lvl (type INTEGER))

)

Οι κενές πλατφόρμες που υπάρχουν στο επίπεδο

(deftemplate levelStat

(slot lvlName (type INTEGER))

(slot num (type INTEGER))

)

## 4.2 Αρχικοποίηση γεγονότων

(deffacts init

(space (name S01) (level 0) (status FULL))

(space (name S02) (level 0) (status EMPTY))

(space (name S03) (level 0) (status FULL))

(space (name S04) (level 0) (status FULL))

(space (name S05) (level 0) (status FULL))

(space (name S06) (level 0) (status FULL))

(space (name S07) (level 1) (status EMPTY));; Οι έξτρα χώροι που βάλαμε

(space (name S08) (level 1) (status FULL))

(space (name S09) (level 1) (status FULL))

(space (name S10) (level 1) (status FULL))

(space (name S11) (level 1) (status FULL))

(space (name S12) (level 1) (status EMPTY))

(space (name S13) (level 2) (status EMPTY))

(space (name S14) (level 2) (status FULL))

(space (name S15) (level 2) (status FULL))

(space (name S16) (level 2) (status FULL))

(space (name S17) (level 2) (status FULL))

(space (name S18) (level 2) (status FULL))

(platform (name P1) (space S01) (status EMPTY))

(platform (name P2) (space S03) (status EMPTY))

(platform (name P3) (space S04) (status EMPTY))

(platform (name P4) (space S05) (status EMPTY))

(platform (name P5) (space S06) (status EMPTY))

(platform (name P6) (space S08) (status EMPTY));; Οι έξτρα πλατφόρμες που βάλαμε

(platform (name P7) (space S09) (status EMPTY))

(platform (name P8) (space S10) (status EMPTY))

(platform (name P9) (space S11) (status EMPTY))

(platform (name P10) (space S14) (status EMPTY))

(platform (name P11) (space S15) (status EMPTY))

(platform (name P12) (space S16) (status EMPTY))

(platform (name P13) (space S17) (status EMPTY))

(platform (name P14) (space S18) (status EMPTY))

(car (plate ZHN3466) (status WAITING))

(car (plate ZZM4556) (status WAITING))

(car (plate IKM7845) (status WAITING))

(car (plate IEN3387) (status WAITING))

(car (plate IXK6876) (status WAITING))

(car (plate IKK2568) (status WAITING))

(car (plate AKK2568) (status WAITING));; Τα έξτρα αυτοκίνητα που περιμένουν

(car (plate IAK2568) (status WAITING))

(car (plate IKA2568) (status WAITING))

(car (plate IKK3568) (status WAITING))

(car (plate IKK2468) (status WAITING))

(car (plate IKK2558) (status WAITING))

(car (plate IKK2566) (status WAITING))

(car (plate IKK2578) (status WAITING))

(car (plate IKK2768) (status WAITING))

(parking (name myParking) (carsWaiting 15) (emptyPlatforms 14))

(east (space S01) (right\_space S02))

(east (space S02) (right\_space S03))

(east (space S04) (right\_space S05))

(east (space S05) (right\_space S06))

(east (space S07) (right\_space S08));; Λόγο τις προσθήκης επιπλέων ορόφων

(east (space S08) (right\_space S09));; Υπάρχουν παραπάνω γεγονότα east

(east (space S10) (right\_space S11))

(east (space S11) (right\_space S12))

(east (space S13) (right\_space S14))

(east (space S14) (right\_space S15))

(east (space S16) (right\_space S17))

(east (space S17) (right\_space S18))

(north (space S01) (up\_space S04))

(north (space S02) (up\_space S05))

(north (space S03) (up\_space S06))

(north (space S07) (up\_space S10));; Λόγο τις προσθήκης επιπλέων ορόφων

(north (space S08) (up\_space S11));; Υπάρχουν παραπάνω γεγονότα north

(north (space S09) (up\_space S12))

(north (space S13) (up\_space S16))

(north (space S14) (up\_space S17))

(north (space S15) (up\_space S18))

(level (space S02) (above\_space S08));; Τα επάνω επίπεδα

(level (space S08) (above\_space S14))

(currentLevel (lvl 0)) ;; Το επίπεδο που προσπαθούμε να γεμίσουμε

(levelStat (lvlName 0) (num 5)) ;; Οι κενές πλατφόρμες σε κάθε επίπεδο

(levelStat (lvlName 1) (num 4))

(levelStat (lvlName 2) (num 5))

)

## 4.3 Σχολιασμός των επιπρόσθετων κανόνων και αλλαγές που έγιναν

### 4.3.1 changeLevel

Ο κανόνας αυτός αυξάνει κατά 1 το επίπεδο όπου προσπαθούμε να γεμίσουμε αν στο επίπεδο που προσπαθούμε να γεμίσουμε (currentLevel). Ο αριθμός του lvlStat είναι 0.

(defrule changeLevel

(declare (salience 20))

?lv <- (currentLevel (lvl ?x))

(levelStat (lvlName ?x) (num 0))

=>

(modify ?lv (lvl (+ ?x 1)))

)

### 4.3.2 swapNorth

Στον κανόνα swapNorth παίρνουμε τον αριθμό του currentLevel και έπειτα αναζητούμε κάποιο γεμάτο χώρο σε αυτό το επίπεδο. Στην συνέχεια, βρίσκουμε ένα γείτονα αυτού και ελέγχουμε αν ο γειτονικός χώρος είναι κενός. Τέλος, χρειαζόσαστε το fact της πλατφόρμας για να αλλάξει χώρο. Αν βγουν αληθή τα προηγούμενα τότε ενημερώνεται ο γεμάτος χώρος πως πλέον είναι άδειος, ενημερώνεται ο άδειος χώρος πως πλέον είναι γεμάτος και ενημερώνεται η πλατφόρμα πως είναι σε άλλο χώρο.

(defrule swapNorth

(currentLevel (lvl ?cLvl))

?fSpace <- (space (name ?sp) (status FULL) (level ?cLvl))

(or (north (space ?sp) (up\_space ?neigb))

(north (up\_space ?sp) (space ?neigb)))

?eSpace <- (space (name ?neigb) (status EMPTY))

?plat <- (platform (space ?sp)

=>

(modify ?fSpace (status EMPTY))

(modify ?eSpace (status FULL))

(modify ?plat (space ?neigb))

)

### 4.3.3 swapEast

Όπως και με την swapNorth, έτσι και με τη swapEast ισχύουν τα ίδια

(defrule swapEast

(currentLevel (lvl ?cLvl))

?fSpace <- (space (name ?sp) (status FULL) (level ?cLvl))

(or (east (space ?sp) (right\_space ?neigb))

(east (right\_space ?sp) (space ?neigb)))

?eSpace <- (space (name ?neigb) (status EMPTY))

?plat <- (platform (space ?sp))

=>

(modify ?fSpace (status EMPTY))

(modify ?eSpace (status FULL))

(modify ?plat (space ?neigb))

)

### 4.3.4 swapLevelE

Αυτός ο κανόνας έχει ως σκοπό την μετακίνηση πλατφόρμας σε άλλο επίπεδο αν είναι κενή η πλατφόρμα. Αναζητούμε κάποιο γεμάτο χώρο σε αυτό το επίπεδο, βρίσκουμε έναν γείτονα αυτού στο επίπεδο, ελέγχουμε αν ο γειτονικός χώρος είναι κενός και αν η πλατφόρμα είναι κενή. Κρατάμε τα facts και τις τιμές για να τα αλλάξουμε. Στη συνέχεια, ενημερώνουμε τον γεμάτο χώρο πως πλέον είναι άδειος, ενημερώνουμε τον άδειο χώρο πως πλέον είναι γεμάτος, ενημερώνουμε την πλατφόρμα πως είναι σε άλλο χώρο, αφαιρούμε κατά 1 το επίπεδο όπου πήγε η πλατφόρμα και προσθέτουμε 1 στο επίπεδο όπου πήγε η πλατφόρμα.

(defrule swapLevelE

?fSpace <- (space (name ?sp) (status FULL) (level ?fsp))

(or (level (space ?sp) (above\_space ?neigb))

(level (above\_space ?sp) (space ?neigb)))

?eSpace <- (space (name ?neigb) (status EMPTY) (level ?esp))

?plat <- (platform (space ?sp) (status EMPTY))

?fls <- (levelStat (lvlName ?fsp) (num ?flvlnum))

?els <- (levelStat (lvlName ?esp) (num ?elvlnum))

=>

(modify ?fSpace (status EMPTY))

(modify ?eSpace (status FULL))

(modify ?plat (space ?neigb))

(modify ?fls (num (- ?flvlnum 1)))

(modify ?els (num (+ ?elvlnum 1)))

)

### 4.3.5 swapLevelF

Αντίστοιχα, με τη swapLevelE η swapLevelF μετακινεί μία γεμάτη πλατφόρμα σε άλλο επίπεδο.

(defrule swapLevelF

?fSpace <- (space (name ?sp) (status FULL))

(or (level (space ?sp) (above\_space ?neigb))

(level (above\_space ?sp) (space ?neigb)))

?eSpace <- (space (name ?neigb) (status EMPTY) )

?plat <- (platform (space ?sp) (status FULL))

=>

(modify ?fSpace (status EMPTY))

(modify ?eSpace (status FULL))

(modify ?plat (space ?neigb))

)

### 4.3.6 carEnters

Αυτός ο κανόνας αφορά την είσοδο του αμαξιού στην πλατφόρμα.

(defrule carEnters

(declare (salience 1000))

?plat <- (platform (name ?platN) (space S02) (status EMPTY))

?car <- (car (status WAITING))

?park <- (parking (carsWaiting ?cw) (emptyPlatforms ?ep))

?stat <- (levelStat (lvlName 0) (num ?numlvl)) ;; Χρειαζόμαστε το fact για να το ενημερώσουμε

=>

(modify ?plat (status FULL))

(modify ?car (status ?platN))

(modify ?park (carsWaiting (- ?cw 1)) (emptyPlatforms (- ?ep 1)))

(modify ?stat (num (- ?numlvl 1))) ;; Μειώνουμε κατά 1 τις διαθέσιμες πλατφόρμες του επιπέδου

)

### 4.3.7 goalFound

Ο κανόνας goalFound αφορά τον στόχο ο οποίος είναι να μην υπάρχουν αυτοκίνητα σε αναμονή. Στην ενδιάμεση κατάθεση είχαμε διαφορετική υλοποίηση.

(defrule goalFound

(declare (salience 10000))

(parking (carsWaiting 0))

=>

(assert (goal found))

(halt)

)

### 4.3.8 howManyCarsWaiting

Αυτός ο κανόνας διαβάζει πόσα αυτοκίνητα περιμένουν να καταγραφούν για αναμονή.

(defrule howManyCarsWaiting

(declare (salience 10000))

(initial-fact)

=>

(printout t "How many cars waiting: ")

(bind ?waits (read))

(assert (waitingCars ?waits))

)

### 4.3.9 registerCar

Αυτός εδώ ο κανόνας καταγράφει πόσα αυτοκίνητα περιμένουν σε αναμονή.

(defrule registerCar

(declare (salience 10000))

?x <- (waitingCars ?n&~0)

=>

(printout t "Plates of the car: ")

(assert (car (plate (read)) (status WAITING)))

(retract ?x)

(assert (waitingCars (- ?n 1)))

)