ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη Μοντέλων Ηθικής Συμπεριφοράς Τεχνητής Νοημοσύνης με Χρήση Deontic Logic

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημήτριος Ν. Ζαφειρακόπουλος

Επιβλέπων: Πέτρος Στεφανέας
Επ. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2018

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Ανάπτυξη Μοντέλων Ηθικής Συμπεριφοράς Τεχνητής Νοημοσύνης με Χρήση Deontic Logic

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Δημήτριος Ν. Ζαφειρακόπουλος

Επιβλέπων: Πέτρος Στεφανέας
Επ. Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 15/11/2018

.....	.....	.....
Πέτρος Σεφανέας	Ανδρέας-Γεώργιος Σταφυλοπάτης	Νικόλαος Παπασπύρου
Επ. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αν. Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2018

……
Δημήτριος Ν. Ζαφειρακόπουλος
Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών ΕΜΠ

Περίληψη

Αυτή η διπλωματική εργασία έχει ως αντικείμενο μελέτης την ανάπτυξη μεθοδολογιών και τυποποίηση μοντέλων που αποσκοπούν στην εξασφάλιση σωστής ηθικά συμπεριφοράς αυτόνομων μονάδων τεχνητής νοημοσύνης με αποδείξιμο τρόπο μέσω χρήσης προτάσεων δεοντικής λογικής. Στο πρώτο μέρος της διπλωματικής εργασίας θα γίνει μια συνοπτική παρουσίαση των βασικών αρχών της δεοντικής λογικής καθώς και στους στόχους που έχει ο χώρος της ηθικής των μηχανών . Στη συνέχεια θα παρουσιαστούν μερικές προσεγγίσεις που αναπτύχθηκαν στα πλαίσια αυτής της διπλωματικής για την εξασφάλιση της επιθυμητής συμπεριφοράς και έπειτα θα παρουσιαστούν συγκεκριμένα μοντέλα , με εστίαση στον ιατρικό χώρο και στο χώρο της αυτοκίνησης, τα οποία αξιοποιώντας τις προαναφερθείσες προσεγγίσεις προσπαθούν να εξασφαλίσουν την επίτευξη των στόχων που έχουν τεθεί με τρόπο μάλιστα που είναι αποδείξιμα συνεπής, σε επίπεδο λογικής τυποποίησης.

Λέξεις κλειδιά

Δεοντική λογική, τεχνητή νοημοσύνη, ηθική των μηχανών, μοντέλο, αντίφαση

Abstract

This thesis has as its focus of study the development of methodologies and the formalization of models that aim at ensuring a correct ethical behavior of autonomous units of artificial intelligence, in a provable way by using deontic logic statements. In the first part, the basic ideas of deontic logic and the goals of machine ethics will be concisely presented. Then some approximations, which aim at ensuring the desirable behavior, that were developed within the thesis will be presented along with some specific models, which focus on the field of medicine and self-driving cars, by utilizing the above mentioned approximations to ensure the satisfaction of the goals that have been set in a way that is provably correct, as far as logical formalization is concerned.

Key words

Deontic logic, artificial intelligence, machine ethics, model, contradiction

Ευχαριστίες

Ολοκληρώνοντας αυτή τη διπλωματική εργασία περατώνονται και οι σπουδές μου ως Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών. Ήταν σίγουρα μια πορεία γεμάτη προκλήσεις, μάθηση και συνεχή προσπάθεια. Το ταξίδι ήταν καθοριστικό για τη διάπλασή μου ,τόσο ως επιστήμονας όσο και ως άνθρωπος γενικότερα. Στα πλαίσια αυτής της πορείας υπήρχαν συγκεκριμένα άτομα που θα ήθελα να ευχαριστήσω.

Το πρώτο άτομο που θέλω να ευχαριστήσω είναι ο επιβλέπων καθηγητής μου στην παρούσα διπλωματική εργασία, κ. Πέτρος Στεφανέας, για τη δυνατότητα που μου έδωσε να εργαστούμε μαζί σε αυτό το ενδιαφέρον θέμα, ιδιαίτερα καθώς με στήριξε και με ενθάρρυνε να προσεγγίσω το αντικείμενο μελέτης από την οπτική που εγώ ο ίδιος επιθυμούσα χωρίς να λείψει στιγμή η βοήθειά του. Ήταν μια εμπειρία που πάντοτε θα την εκτιμώ.

Επίσης ευχαριστώ τους υπόλοιπους καθηγητές της σχολής με τους οποίους συνεργάστηκα σε όλη τη διάρκεια των σπουδών μου στο Εθνικό Μετσόβιο Πολυτεχνείο.

Σε πιο προσωπικό επίπεδο έχω την ειλικρινή ανάγκη να ευχαριστήσω τους συγγενείς και φίλους που με στήριξαν σε αυτήν τη δύσκολη πορεία, καθώς χωρίς την παρουσία τους δεν θα είχα καταφέρει να συνεχίσω.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

Στο πρώτο μέρος της διπλωματικής εργασίας, θα παρουσιαστεί μια θεμελίωση των θεωρητικών βάσεων που χρειάστηκαν για την ανάπτυξη των διάφορων τμημάτων της έρευνας που έγινε συνολικά στα πλαίσια της εργασίας.

1.1 Δεοντική Λογική [1][2]

Επιλέχτηκε χρήση προτάσεων deontic logic (δεοντικής λογικής) γιατί αποσκοπώ σε μια, κατά το δυνατόν, μαθηματική θεμελίωση των μοντέλων και αλγορίθμων που αναπτύχθηκαν. Ακολουθεί παρουσίαση των αρχών deontic logic που χρησιμοποιήθηκαν.

Για την κατανόηση των εννοιών deontic logic είναι χρήσιμο να παρουσιαστεί πρώτα ένα γενικότερο θέμα, αυτό των εννοιών modal logic.

1.1.1 Τροπική Λογική

Ένα modal(τροπικό) είναι μια έκφραση (όπως απαραίτητα ή πιθανώς) που χρησιμοποιείται για να αξιολογήσει την αλήθεια μιας κρίσης. Η modal logic (τροπική λογική) είναι η μελέτη της συμπεριφοράς εκφράσεων όπως είναι αναγκαίο, ή πιθανό ότι. Όμως, ο όρος modal λογική μπορεί να χρησιμοποιηθεί πιο γενικά για μια οικογένεια σχετικών συστημάτων. Αυτά περιλαμβάνουν λογικές για πιστή, για χρονικές εκφράσεις, deontic εκφράσεις όπως είναι απαραίτητο ή επιτρεπτό ότι και άλλα πολλά. Η κατανόηση της modal λογικής είναι ιδιαίτερα πολύτιμη στην τυπική ανάλυση φιλοσοφικών επιχειρημάτων όπου εκφράσεις τις οικογένειας modal είναι και συνήθεις και περίπλοκες. Επίσης, η λογική modal έχει πολλές εφαρμογές στην επιστήμη των υπολογιστών.
Σε στενά πλαίσια, η μελέτη της λογικής modal μελετά τη λογική που περιλαμβάνει χρήση των εκφράσεων απαραίτητα και πιθανά. Όμως, ο όρος modal λογική μπορεί να χρησιμοποιηθεί γενικότερα για μια οικογένεια κανόνων και μια ποικιλία συμβόλων.

Λογική	**Σύμβολα**	**Εκφράσεις που Περιγράφονται**
Τροπική Λογική	□	Είναι αναγκαίο ότι ..
	◊	Είναι πιθανό ότι …
Δεοντική Λογική	*O*	Είναι υποχρεωτικό ότι …
	P	Είναι επιτρεπτό ότι …
	F	Είναι απαγορευμένο ότι …
Χρονική Λογική	*G*	Πάντα θα ισχύει ότι …
	F	Θα ισχύσει ότι …
	H	Πάντοτε ίσχυε ότι …
	P	Ίσχυε ότι …
Δοξαστική Λογική	*Bx*	*O x* πιστεύει ότι …

Οι πιο γνωστές λογικές της οικογένειας modal κατασκευάζονται από μια "αδύναμη" λογική, γνωστή ως K. Όπως είδαμε , αφορούν ανάγκη και πιθανότητα και υπάρχει μια ποικιλία διαφορετικών συστημάτων με την K ως θεμέλιο. Τα σύμβολα περιλαμβάνουν '~' για 'όχι/μη', '→' για 'αν…τότε', και '□' για τον modal τελεστή 'είναι απαραίτητο ότι'. (Τα '&', '∨', και '↔' προκύπτουν εκ των '~' και '→' όπως στον προτασιακό.)

Necessitation Rule: Αν το *A* είναι θεώρημα του **K**, τότε είναι και το □*A*. (Κανόνας Αναγκαιότητας)
Distribution Axiom: □(*A*→*B*) → (□*A*→□*B*). (Αξίωμα Κατανομής)

Με βάση τον κανόνα αναγκαιότητας, κάθε θεώρημα είναι αναγκαίο. Το δεύτερο, λέει πως αν είναι αναγκαίο το αν Α τότε Β , τότε αν αναγκαίο Α, αναγκαίο και Β. Ο τελεστής για το πιθανώς ορίζεται από του απαραιτήτου, θέτοντας ◊*A* = ~□~*A* . Στο K, οι τελεστές απαραιτήτου και αναγκαίου συμπεριφέρονται σαν τους ποσοδείκτες υπάρχει και για κάθε. Για παράδειγμα ο ορισμός του ◊ από □ αντιστοιχεί στην αναλογία ∀*xA* με ~∃*x*~*A* στον κατηγορηματικό λογισμό. Επίσης, □(*A*&*B*) περιέχει □*A*&□*B* και αντιστρόφως, ενώ □*A*∨□*B* ενέχει □(A∨*B*), αλλά όχι αντιστρόφως. Αυτό θυμίζει το ότι ∀*x*(*A*&*B*) περιέχει ∀*xA*&∀*xB* και αντιστρόφως, αλλά ∀*xA* ∨ ∀*xB* περιέχει ∀*x*(*A* ∨ *B*) όχι όμως αντιστρόφως.
Το K είναι πολύ αδύναμο για να μας δώσει μια επαρκή περιγραφή της αναγκαιότητας. Το παρακάτω αξίωμα δεν είναι αποδείξιμο στο K αλλά είναι ξεκάθαρα επιθυμητό

(*M*) □*A*→*A*

Το M μας λέει ότι ότι είναι αναγκαίο ισχύει. Παρατηρούμε ότι το M θα ήταν λάθος αν το σύμβολο □ σημαίνει οφείλει ότι. Έτσι, το M ξεχωρίζει τις λογικές αναγκαιότητας από άλλες modal λογικές. Μια βασική modal λογική δημιουργείται προσθέτοντας το M στο K (κάποιοι το λένε T)

Πολλοί μελετητές της λογικής υποστηρίζουν ότι το M είναι ακόμα αδύναμο για να τυποποιήσει τη λογική ανάγκης και πιθανότητας και εισάγουν επιπλέον αξιώματα .

(4) $\Box A \rightarrow \Box\Box A$
(5) $\Diamond A \rightarrow \Box\Diamond A$

Το S4 προκύπτει προσθέτοντας το (4) στο M. Αντίστοιχα, το S5 με το (5). Στο S4 η πρόταση□□A αντιστοιχεί στην □A. Ως αποτέλεσμα, κάθε ακολουθία □ μπορεί να αντικατασταθεί με ένα □ και το ίδιο ισχύει και για ρόμβους. Αυτό οδηγεί στην ιδέα ότι οι modal τελεστές σε παράθεση είναι περιττοί. Λέγοντας ότι το A είναι απαραιτήτως απαραίτητο έχουμε πλεονασμό. Το σύστημα S5 έχει ισχυρότερες αρχές απλοποιήσεις modal τελεστών. Στο S4 μια ακολουθία modal τελεστών του ίδιου είδους απλοποιούνται από ένα τελεστή, ενώ στο S5 , κουτιά και διαμάντια απλοποιούνται από τον τελευταίο της ακολουθίας.

S4: □□…□ = □ και ◊◊…◊ = ◊
S5: 00…□ = □ και 00…◊ = ◊, όπου κάθε 0 είναι είτε □ ή ◊

Μπορεί κάνεις να διαφωνήσει για την ορθότητα αυτών ή άλλων αρχών σχετικά με τα σύμβολα κουτιού ή ρόμβου. Αλλά, αυτές μπορούν να αρθούν αναγνωρίζοντας ότι οι λέξεις πιθανά και αναγκαία έχουν πολλές διαφορετικές χρήσεις. Έτσι, η αποδοχή ή όχι των αξιωμάτων βασίζεται σε ποιες χρήσεις έχουμε κατά νου. Για αυτό το λόγο, δεν υπάρχει μια λογική modal αλλά μια ολόκληρη οικογένεια συστημάτων περί του M.
Το σύστημα B φτιάχνεται προσθέτοντας το ακόλουθο αξίωμα στο M.

(*B*) $A \rightarrow \Box\Diamond A$

Είναι ενδιαφέρον να δούμε ότι το S5 φτιάχνεται επίσης προσθέτοντας το B στο S4. Το αξίωμα B εγείρει ένα σημαντικό προβληματισμό για την ερμηνεία των modal τύπων. Το B λέει ότι αν A ισχύει τότε το A είναι αναγκαίως πιθανό. Θα μπορούσαμε να πούμε ότι το B πρέπει πάντα να υιοθετείται σε κάθε modal λογική, αφού οπωσδήποτε, αν το A ισχύει τότε το A είναι αναγκαία πιθανό. Όμως, υπάρχει πρόβλημα με αυτό τον ισχυρισμό δείχνοντας ότι το $\Diamond\Box A \rightarrow A$ *είναι* αποδείξιμο από το B. Έτσι το $\Diamond\Box A \rightarrow A$ *πρέπει* να είναι αποδεκτό αν το B είναι αποδεκτό. Όμως το $\Diamond\Box A \rightarrow A$ λέει ότι αν το A είναι πιθανώς αναγκαίο τότε το A ισχύει και αυτό μόνο προφανές δεν είναι. Γιατί το B φαίνεται προφανές, ενώ κάτι που συνεπάγεται δεν είναι? Η απάντηση βρίσκεται στην επικίνδυνη αμφισημία της γλωσσικής ερμηνείας του $A \rightarrow \Box\Diamond A$. Συχνά χρησιμοποιούμε την έκφραση, εάν A τότε απαραίτητα B για να πούμε ότι η σχέση "Αν A τότε B" είναι απαραίτητη. Αυτό αντιστοιχεί στο $\Box(A \rightarrow B)$. Σε άλλες περιπτώσεις εννοούμε ότι αν A, τότε το B αναγκαία ισχύει. $A \rightarrow \Box B$. Στη γλώσσα το αναγκαία είναι επίρρημα και και επειδή τα επιρρήματα θέτονται κοντά σε ρήματα δεν έχουμε φυσικό τρόπο να δείξουμε αν ο modal τελεστής αφορά τη συνολική

συνθήκη ή το επακόλουθό της. Για τους ίδιους λόγους μπορεί να μπερδέψουμε το Β $A \rightarrow \Box\Diamond A$ *με*

$\Box(A \rightarrow \Diamond A)$. Αλλά το $\Box(A \rightarrow \Diamond A)$ δεν είναι το ίδιο με το Β, γιατί το $\Box(A \rightarrow \Diamond A)$ είναι ηδη θεώρημα του Μ αλλά το Β όχι. Ένας απλός τρόπος να προστατευθούμε από αυτό είναι να εκφράσουμε το Β ως $\Diamond\Box A \rightarrow A$, *όπου* ο κίνδυνος αμφισημίας είναι μειωμένος.

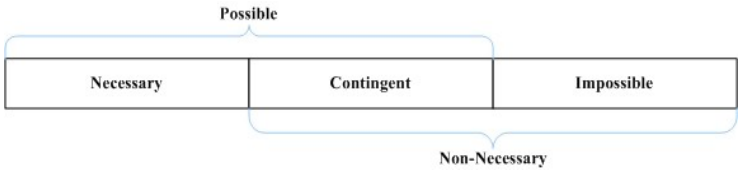
Η αληθετική modal λογική είναι, σε γενικές γραμμές η λογική της αναγκαίας αλήθειας και των σχετικών εννοιών. Οι παρακάτω είναι οι πέντε βασικές "καταστάσεις"

it is necessary (necessarily true) that- (□) αναγκαίο ότι
it is possible that (◊) -πιθανό ότι
it is impossible that -αδύνατο
it is non-necessary- that μη αναγκαίο
it is contingent that -πιθανώς να αλλάξει

Παρόλο που όλοι οι πέντε παραπάνω τελεστές είναι ορίσιμοι με βάση τους τέσσερις πρώτους, συνήθως χρησιμοποιούμε αυτόν της ανάγκης και οι υπόλοιποι προκύπτουν ως εξής.

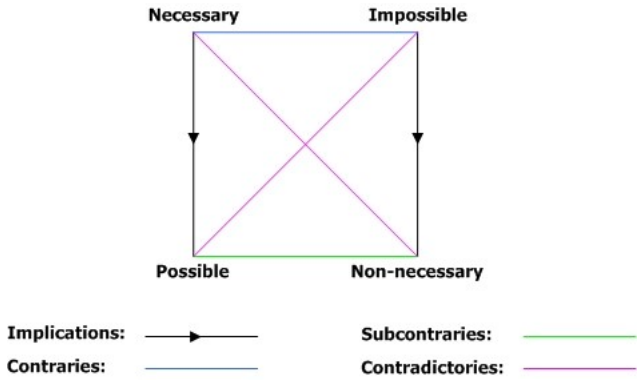
It is possible that p ($\Diamond p$) $=_{df} \sim\Box\sim p$
It is impossible that p $=_{df} \Box\sim p$
It is non-necessary that p $=_{df} \sim\Box p$
It is contingent that p $=_{df} \sim\Box p \ \& \ \sim\Box\sim p$

Οι όροι γίνονται πιο κατανοητοί με το παρακάτω γράφημα.



Τα τρία ορθογώνια κυτία είναι εξαντλητικά και αμοιβαίως ξεχωριστά. Κάθε πρόταση είναι είτε απαραίτητη, είτε πιθανή, είτε απίθανη αλλά καμία δεν μπορεί να είναι παραπάνω από μια από αυτές. Οι πιθανές προτάσεις είναι αυτές που είναι είτε αναγκαίες, είτε πιθανές να αλλάξουν και οι μη απαραίτητες είναι αυτές που μπορούν να αλλάξουν (ως προς το αν ισχύουν) ή απίθανες.

Και το παρακάτω γράφημα βοηθά στην εξήγηση.



1.1.2 Εμβάθυνση σε Δεοντική Λογική

Η deontic λογικές εισάγουν το σύμβολο OB, για το είναι υποχρεωτικό, από το οποίο τα σύμβολα P ,επιτρέπεται, και F είναι απαγορευμένο ορίζονται ως $PA = \sim OB\sim A$ και $FA = OB\sim A$. Το deontic ανάλογο του modal αξιώματος (*M*): $OBA \rightarrow A$ *είναι* ξεκάθαρα μη απαραίτητο για την deontic λογική. (Διότι δυστυχώς, ό,τι είναι υποχρεωτικό δεν ισχύει και πάντα.) Όμως, μπορούμε να φτιάξουμε ένα πρωτόγονο σύστημα D deontic λογικής προσθέτοντας το (D) στο K.

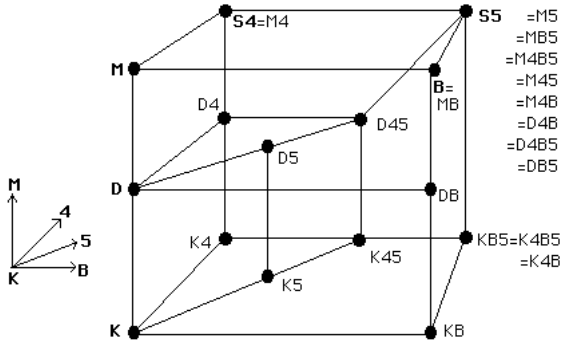
(*D*) $OBA \rightarrow PA$

Το (*D*) εγγυάται τη συνέπεια του συστήματος υποχρεώσεων διότι με αυτό αν το A είναι απαραίτητο, τότε είναι και επιτρεπτό. Ένα σύστημα που μας υποχρεώνει να πετύχουμε το A αλλά δεν μας το επιτρέπει, μας φέρνει προ αδιέξοδου. Παρόλο που κάποιοι θα έλεγαν πως τέτοιες συγκρούσεις υποχρεώσεων είναι πιθανές, οι περισσότεροι μελετητές της λογικής αποδέχονται το (*D*).

$OB(OBA \rightarrow A)$ είναι ένα ακόμα deontic αξίωμα που φαίνεται επιθυμητό. Παρόλο που είναι λάθος να πούμε ότι αν το A είναι υποχρεωτικό τότε το A ισχύει, $(OBA \rightarrow A)$, αυτή η συνθήκη οφείλει να ισχύει. Έτσι κάποιοι μελετητές της λογικής πιστεύουν ότι το *D πρέπει* να ενισχυθεί με το $OB(OBA \rightarrow A)$.

Προβληματισμοί περί της επανάληψης τελεστών εμφανίζονται και πάλι στη deontic λογική. Σε κάποιες οπτικές περί υπερχρέωσης, OBOBA είναι απλώς OBA. Οφείλει να ισχύει ότι οφείλει να ισχύει θεωρείται σχεδόν τραύλισμα, το έξτρα οφείλει δεν προσθέτει τίποτα νέο. Έτσι αξιώματα προσθέτονται για να βεβαιώσουν την αντιστοιχία OBOBA και OBA. Η πιο γενική πολιτική ακολουθίας που φαίνεται στο S5 μπορεί επίσης να υιοθετηθεί. Ωστόσο, υπάρχουν εναλλακτικές ερμηνείες στις οποίες OBA και OBOBA διαφοροποιούνται. Η ιδέα είναι ότι υπάρχουν διαφορές μεταξύ των οφειλών που όντως έχουμε και αυτών που θα έπρεπε να υιοθετήσουμε. Για παράδειγμα το οφείλει να ισχύει ότι οφείλει να ισχύει A οδηγεί σε υιοθεσία κάποιων υποχρεώσεων που μπορεί να μην ισχύουν ακόμα με το αποτέλεσμα OBOBA να ισχύει όταν OBA δεν ισχύει.

Παραθέτεται παρακάτω ένα διάγραμμα που δείχνει τις σχέσεις μεταξύ των modal logics,που προκύπτουν προσθέτοντας τα (*D*), (*M*), (4), (*B*) and (5) στο **K**.



Deontic Logic

Η deontic λογική είναι το κομμάτι της συμβολικής λογικής που ασχολείται κυρίως με τις παρακάτω έννοιες

permissible (permitted)	must
impermissible (forbidden, prohibited)	supererogatory (beyond the call of duty)
obligatory (duty, required)	indifferent / significant
omissible (non-obligatory)	the least one can do
optional	better than / best / good / bad
ought	claim / liberty / power / immunity

Κάποιες, βέβαια από αυτές τις έννοιες λαμβάνουν περισσότερη προσοχή από άλλες. Όμως, όλοι όσοι δουλεύουν σε αυτό τον τομέα θα μπορούσαν να δουν πως συστήματα που αφορούν σε μοντέλα της συμβολής αυτών των εννοιών είναι όντως τμήματα της deontic λογικής.

Ως ενότητα συμβολικής λογικής , η λογική deontic είναι θεωρητικού ενδιαφέροντος για τους ίδιους λόγους που και η modal λογική είναι θεωρητικού ενδιαφέροντος. Όμως, παρόλο που χρειάζεται να είμαστε προσεκτικοί όταν επιχειρούμε να συνδέσουμε τη deontic λογική και την πρακτικότητα, πολλές από τις σχετικές ιδέες χρησιμοποιούνται εν τούτοις σε μια προσπάθεια να ρυθμίσουμε την ζωή μας (και να αξιολογήσουμε διάφορα ζητήματα). Για αυτούς τους λόγους οι λογικές deontic αφορούν άμεσα θέματα ιδιαίτερης πρακτικής σημασίας, όπως η ηθική , ο νόμος, οργανισμοί κοινωνικού ή επιχειρησιακού χαρακτήρα και συστήματα ασφαλείας. Και, επειδή αυτά έχουν πρακτική αξία, έχει κατ' επέκταση και η λογική deontic.

Η deontic λογική έχει επηρεαστεί εντόνως από ιδέες της λογικής modal. Αναλογίες μεταξύ αληθετικής modal λογικής και ιδεών σχετικών με το χώρο της deontic λογικής παρατηρήθηκαν από τον 14ο αιώνα, όπου μπορούμε να πούμε ότι κάποια θεμέλια της σύγχρονης deontic λογικής πρωτοεμφανίστηκαν. Παρόλο που άτυπο ενδιαφέρον στη μελέτη κάποιων τμημάτων της deontic λογικής συνεχίστηκε, η τάση να μελετάμε λογική με συμβολικές και ακριβείς τεχνικές των μαθηματικών καθιερώθηκε στον 20ο αιώνα με τον von Wright , τη κεντρική αρχική φιγούρα στην εμφάνιση της deontic λογικής ως πλήρως αναπτυγμένο κομμάτι της συμβολικής λογικής του 20ου αιώνα. Κρίνεται σκόπιμο να παρουσιαστούν κάποια πιο γενικά χαρακτηριστικά της deontic λογικής, σε διαισθητικό επίπεδο, πριν περάσουμε σε τυποποιήσεις.

Πριν τον von Wright και την καθιέρωση της deontic λογικής, ως ακαδημαϊκού τομέα ερευνών, πρέπει να σημειωθεί ότι υπήρχε ένα σημαντικό σημείο, με τον Μαλλυ το 1926, το οποίο δεν είχε τελικά την επιρροή στην συμβολική deontic λογική που θα μπορούσε να έχει.

Παρόλα αυτά ,το σύστημα που όρισε τον καθιστά σημαντικό θεμελιωτή στη deontic λογική. Ο Μαλλυ δεν επηρεάστηκε από πρώιμες εξελίξεις της αληθετικης modal λογικής. Αυτό έρχεται σε αντίθεση με την ύστερη τάση το 1950 όταν η deontic λογική επανεμφανίστηκε, ως πλήρης ακαδημαϊκός τομέας και βαθιά επηρεασμένη από την αληθετικη modal λογική. Ο Μαλλυ ήταν ο πρώτος που όρισε τη deontic λογική με τη σύνταξη βασισμένη στον προτασιακό λογισμό, μια στρατηγική που στην οποία οι υπόλοιποι γρήγορα στράφηκαν μετά από μια περίοδο απόκλισης -υστέρα από το πρώιμο έργο του von Wright. Ο Μάλλυ ήταν ο πρώτος που χρησιμοποίησε σταθερές στη deontic λογική. Επίσης, ήταν ο πρώτος που έκανε μια καταγραφή non-conditional (μη εξαρτημένων) και conditional (εξαρτημένων) προτάσεων υπερχρέωσης, η οποία πρόσφερε μια ανάλυση των υποχρεώσεων μέσω ενός ‘εναδικού’ deontic τελεστή σε συνδυασμό με μια σταθερά (που θύμιζε ανεπιτυχείς προσπάθειες του von Wright)

Μέχρι τώρα παρουσιάστηκε η απαραίτητη θεωρία που αφορά την modal αναγκαιότητα. Τώρα, η εργασία περνά στο κυριότερο κορμό της απαραίτητης θεωρίας που είναι η deontic λογική.

Οι πέντε “κανονιστικές” καταστάσεις στο παραδοσιακό σχήμα για deontic λογική είναι

είναι υποχρεωτικό (**OB)**
είναι επιτρεπτό (**PE**)
είναι μη επιτρεπτό (**IM**)
είναι παραλείψιμο (**OM**)
είναι προαιρετικό(**OP)**

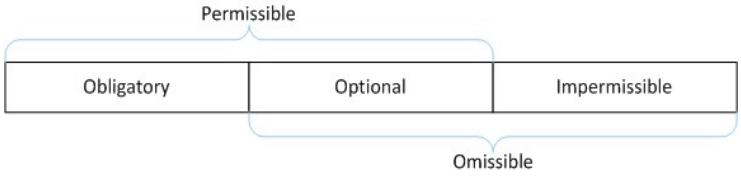
Τα πρώτα τρία είναι τα βασικότερα, αλλά το τέταρτο γενικώς αγνοείται και το πέμπτο εξισώνεται με “είναι αδιάφορο ότι ρ” (όπως ορίζεται με βάση τα τρία πρώτα) που δεν είναι κομμάτι το παραδοσιακού σχήματος. Γενικώς, το ένα από τα δυο πρώτα καθορίζεται ως το

βασικό και τα υπόλοιπα ορίζονται με βάση αυτό, αν και θα μπορούσαμε να πάρουμε οποιοδήποτε από τα τέσσερα πρώτα.

PE*p* ↔ ~**OB**~*p*
IM*p* ↔ **OB**~*p*
OM*p* ↔ ~**OB***p*
OP*p* ↔ (~**OB***p* & ~**OB**~*p*)

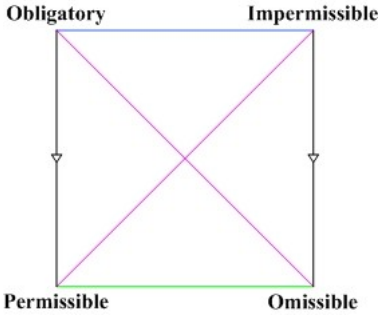
Αυτά δείχνουν ότι κάτι είναι επιτρεπτό, αν και μόνο αν η άρνηση του δεν είναι υποχρεωτική, μη επιτρεπτό αν η άρνηση του είναι υποχρεωτική, παραλείψιμο αν δεν είναι υποχρεωτικό και προαιρετικό αν και μόνο αν (ανν) ούτε αυτό ούτε η άρνηση του είναι υποχρεωτικά. Αυτό ονομάζεται "The Traditional *Definitional* Scheme (TDS)" (Παραδοσιακό Σχήμα Ορισμού). Αν ξεκινούσαμε μόνο με το OB και χρησιμοποιούσαμε τους τύπους στα δεξιά θα μπορούσαμε να οδηγηθούμε στα υπόλοιπα. Παρόλο που υπάρχουν άλλες εναλλακτικές, αυτές είναι ιδιαίτερα "φυσικές" και χρησιμοποιούνται ευρέως. Τώρα, αν ανατρέξουμε στα προηγούμενα περί αληθετικής modal λογικής είναι ξεκάθαρο ότι εκείνο το σχήμα ορισμού είναι ανάλογο σε αυτό το deontic. Η διαφορά είναι καθαρά "συντακτική", αλλάζουμε το **OB** με □, **PE** με ◊, κλπ

Επίσης πέραν του TDS, φαίνεται να ισχύει και το παρακάτω τρίπτυχο. (Traditional Threefold Classification (TTC)) (Παραδοσιακή Τρίπτυχη Κατηγοριοποίηση)

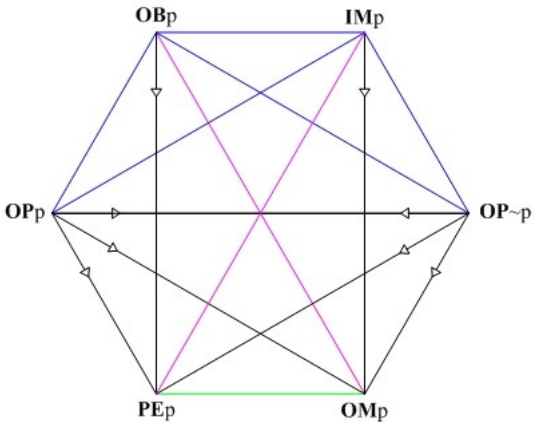


Και εδώ, όλες οι προτάσεις χωρίζονται σε τρεις εξαντλητικές και αμοιβαία ξεχωριστές κλάσεις. Κάθε πρόταση είναι υποχρεωτική, προαιρετική, είτε ανεπίτρεπτη, αλλά καμία δεν είναι σε παραπάνω από μια κατηγορίες. Επίσης, οι επιτρεπτές προτάσεις είναι αυτές που είναι είτε υποχρεωτικές είτε προαιρετικές και οι παραλείψιμες είναι αυτές που είναι είτε ανεπίτρεπτες ή προαιρετικές. Φυσικά, και πάλι υπάρχει πλήρης αναλογία με την αντίστοιχη τρίπτυχη κατηγοριοποίηση των αληθετικών modal προτάσεων.

(Deontic Square -DS) (Δεοντικό Τετράγωνο)



Οι λογικοί τελεστές στις γωνίες ερμηνεύονται αντίστοιχα με το τους ανάλογους στο τετράγωνο για modal λογική. Ακολουθεί ένα ακόμα διάγραμμα.



Με αυτές τις αναλογίες, δεν προκαλεί έκπληξη που ο βασικός μας τελεστής σε deontic logic "είναι υποχρεωτικό ότι " (OB), αναφέρεται συχνά ως δεοντική αναγκαιότητα. Όμως υπάρχουν και σημεία στα οποία σταματούν οι αναλογίες. Οι παρακάτω αρχές είναι κομμάτι της αληθετικής modal λογικής

If □*p* then *p* (if it is necessary that *p*, then *p* is true).
If *p* then ◊*p* (if *p* is true, then it is possible).

Τα ανάλογα σε deontic λογική είναι
If **OB**p then *p* (if it is obligatory that *p*, then *p* is true).
If *p* then **PE***p* (if *p* is true, then it is permissible).

Αυτά βέβαια είναι ξεκάθαρα λάθος, διότι υποχρεώσεις μπορούν να παραβιαστούν και επίσης, πράγματα που δεν επιτρέπονται, συμβαίνουν.
Όμως, καθώς οι διάφοροι ερευνητές εξερευνούσαν διάφορες γενικεύσεις της αληθετικής modal λογικής, άρχισαν να επεξεργάζονται κλάσεις, στις οποίες οι τελεστές αναγκαιότητας, δεν συνεπάγονται αλήθεια/ισχύ. Αυτό οδήγησε περισσότερους στο να θεωρούν τη deontic

αναγκαιότητα και κατά συνέπεια τη deontic λογική τμήμα της modal λογικής.

1.1.2.1 Πρώτες Θεμελιώσεις

Είναι γενικώς βολικό να προσθέσουμε περισσότερη συστηματική οργάνωση. Υποθέτοντας ότι έχουμε το γνωστό προτασιακό λογισμό με άπειρες μεταβλητές ($P_1,\ldots,P_n,\ldots$) με τελεστές αληθείας (~ and →) και τον deontic τελεστή OB.

Εκτός αν πούμε κάτι διαφορετικό , μας ενδιαφέρουν οι deontic λογικές που εμπεριέχουν τον κλασσικό προτασιακό λογισμό. Έτσι, βάζουμε αυτό ως πρώτο "συστατικό" για να ορίσουμε τη deontic λογική, ώστε το **OB***p* → ~~**OB***p* να μπορεί να αντληθεί από το σύστημα.

Παραπάνω, ορίζοντας το Traditional Definitional Scheme είδαμε, ότι οποιαδήποτε από τις τέσσερις πρώτες από τις πέντε βασικές αρχές και να παίρναμε θα μπορούσαμε να ορίσουμε τις υπόλοιπες με βάση αυτής. Έτσι, θα θέλαμε να μπορούμε να φτιάξουμε τις συνεπαγωγές του σχήματος που αποφασίσαμε, με την OB ως βασική. Αλλά μέχρι στιγμής δεν μπορούσαμε. Για παράδειγμα, είναι προφανώς επιθυμητό να έχουμε το **OB***p* → ~**PE**~*p* ως θεώρημα. Αλλά, δεν είναι εξαγώγιμο ακόμα.

Το **OB***p* → ~**PE**~*p* *είναι* ξεκάθαρα αντίστοιχο του **OB***p* → ~~**OB**~~*p* το οποίο από προτασιακό εκπίπτει σε **OB***p* → **OB**~~*p* , άλλα αυτός ο τύπος δεν είναι ταυτολογία, όποτε η απόδειξη δε θεωρείται ολοκληρωμένη. Μέχρι στιγμής έχουμε deontic wffs (καλώς ορισμένους τύπους) και προτασιακό λογισμό αλλά όχι προτασιακή λογική. Για αυτό χρειαζόμαστε ξεκάθαρες αρχές για τον τελεστή και συγκεκριμένα για να μπορούμε να φτιάξουμε τις εναλλακτικές συνεπαγωγές που θίξαμε παραπάνω. Χρειαζόμαστε ένα χαρακτηριστικό των κλασικών modal λογικών.

(Chellas 1980):

OB-RE: If $p \leftrightarrow q$ is a theorem, then so is **OB***p* ↔ **OB***q*.

Αυτός ο κανόνας μας λέει (χοντρικά) ότι αν δυο τύποι είναι αποδείξιμα ισοδύναμοι, τότε το αποτέλεσμά του να βάλεις το OB μπροστά απ'τον καθένα, είναι εξίσου ισοδύναμοι. Με αυτό και το Traditional Definitional Scheme μπορούμε να δείξουμε τις ισοδυναμίες στα εναλλακτικά σχήματα. Για παράδειγμα, αφού ⊢ $p \leftrightarrow$ ~~*p*, από **OB**-RE, έχουμε ⊢ **OB***p* ↔ **OB**~~*p*, ⊢ **OB***p* ↔ ~~**OB**~~*p*, που μας δίνει⊢ **OB***p* ↔ ~**PE**~*p,* με βάση το σχήμα μας.

Όλα τα συστήματα εδώ θα περιέχουν το RE (είτε ως βασικό , είτε μέσω επαγωγών). Θα περιέχουν επίσης (έκτος αν πούμε διαφορετικά) μια ακόμη αρχή, μια θέση που λέει ότι μια λογική αντίφαση (συμβατικά τη δείχνουμε "⊥") είναι πάντοτε παραλείψιμη.

OD: ~**OB**⊥

Για παράδειγμα, το OD δείχνει ότι είναι λογική αλήθεια ότι δεν είναι υποχρεωτικό ότι οι φόροι μου πληρώνονται και δεν πληρώνονται. Παρόλο που το OD δεν είναι αποδεκτό από όλους χωρίς διαφωνίες, είναι δυνατό να το αποδεχτούμε και όπως το RE έχει εισαχθεί κατά πολλοίς στη deontic λογική. Θα επικεντρωθούμε σε συστήματα που αποδέχονται και το RE και το OD .

Τυπική έκφραση Threefold Classification, και Deontic Square of Opposition (Δεοντικό Τετράγωνο Αντίθεσης)

DS: (**OB***p* ↔ ~**OM***p*) & (**IM***p* ↔ ~**PE***p*) & ~(**OB***p* & **IM***p*) & ~(~**PE***p* & ~**OM***p*) & (**OB***p* → **PE***p*) & (**IM***p* → **OM***p*).

TTC: (**OB***p* ∨ **OP***p* ∨ **IM***p*) & [~(**OB***p* & **IM***p*) & ~(**OB***p* & **OP***p*) & ~(**OP***p* & **IM***p*)].

Τελικά από Traditional Definitional Scheme, τα DS και TTC είναι ταυτολογικά ανάλογα στο ότι οι υποχρεώσεις δεν μπορούν να έρχονται σε σύγκρουση.

NC: ~(**OB***p* & **OB**~*p*)

Έτσι το Traditional Scheme βασίζεται στην ορθότητα του NC (και στους παραδοσιακούς ορισμούς των τελεστών). Όντως, το Traditional Scheme δεν είναι τίποτε άλλο από κεκαλυμμένη μορφή του NC.

Το NC δεν πρέπει να μπερδευτεί από άποψη περιεχομένου με την αρχή που αναφέραμε προηγούμενα ,OD (~**OB**⊥). Το OD δείχνει ότι καμία λογική αντίφαση δεν μπορεί να είναι υποχρεωτική, ενώ το NC μας λέει ότι δεν μπορεί να υπάρχουν δυο διαφορετικές προτάσεις, που και οι δυο είναι υποχρεωτικές, ενώ η μια είναι η άρνηση της άλλης. Η αποδοχή ή μη του NC είναι που οδηγεί σε έναν από τους κυριότερους διχασμούς των συστημάτων deontic λογικής Θα δούμε παρακάτω ένα από τα συστήματα που το αποδέχονται, το "Standard Deontic Logic" (Κανονική Δεοντική Λογική) .

1.1.2.2 Κανονική Δεοντική Λογική

"Standard Deontic Logic (Κανονική Δεοντική Λογική)" (SDL) είναι η πιο συνήθης μορφή deontic λογικής και μια από τις πρώτες που θεμελιώθηκαν με αξιώματα. *Στηρίζεται* στον προτασιακό λογισμό και στην πραγματικότητα είναι ένα μέλος της πιο μελετημένης κλάσης modal λογικής, της "κανονικής modal λογικής". Είναι εναδική, διότι ο βασικός τελεστής της είναι μιας θέσης (όπως το ~, και όχι, πχ δυο , όπως το →).
Υποθέτουμε ότι έχουμε μια γλωσσά κλασσικού προτασιακού υπολογίσιμου, με ένα άπειρο σύνολο μεταβλητών, τους τελεστές ~ και →, και τον τελεστή **OB**. Το SDL "αξιωματικοποιήται" ως έξης

SDL:	A1. Όλες οι ταυτολογίες	(TAUT)
	A2. **OB**(*p* → *q*) → (**OB***p* → **OB***q*)	(**OB**-K)
	A3. **OB***p* → ~**OB**~*p*	(**OB**-D)
	R1. Αν ⊢ *p* και ⊢ *p* → *q* τότε ⊢ *q*	(MP)
	R2. Αν ⊢ *p* τότε ⊢ **OB***p*	(**OB**-NEC)

Το SDL είναι απλά η κανονική modal λογική "D" η "KD" με την κατάλληλη ερμηνεία (για deontic ζητήματα).

Το TAUT είναι στάνταρ για τα κανονικά συστήματα modal λογικής. Το OB-K που είναι το K αξίωμα, παρόν σε όλες τις modal λογικές, μας λέει ότι αν ένα σε μια σχέση υπόθεσης-συμπεράσματος το σύνολο είναι υποχρεωτικό και το υποθετικό κομμάτι είναι υποχρεωτικό, τότε είναι και το συμπερασματικό. Το **OB**-D μας λέει ότι το *p είναι* υποχρεωτικό αν και μόνο αν η άρνηση του δεν είναι. Πρακτικά, εννοούμε "όχι αντιφάσεις" αλλά, λέγεται D (deontic) στις κανονικές modal λογικές. Το MP είναι το Modus Ponens και μας λέει ότι αν μια σχέση συνεπαγωγής και το πρώτο μέλος της είναι θεωρήματα, τότε και το συμπερασματικό. είναι.

Το TAUT με το MP μας δίνουν την πλήρη ισχύ του προτασιακού λογισμού (και συχνά αναφερόμαστε σε αυτό ως PC). Όπως είδαμε νωρίτερα, το PC δεν έχει από μόνο του deontic κομμάτι. Το **OB**-NEC μας λέει ότι αν οτιδήποτε είναι θεώρημα, τότε η πρόταση ότι αυτό είναι και υποχρεωτικό είναι επίσης θεώρημα. Σημειώνουμε ότι αυτό συνεπάγεται ότι κάτι θα είναι πάντοτε υποχρεωτικό (έστω και αν αυτό είναι μόνο οι λογικές αλήθειες). Κάθε μια από τις deontic αρχές **OB**-K, **OB**-D, και **OB**-NEC δέχονται αμφισβήτηση από κάποιους. Όμως, για να αποφευχθεί η σύγχυση σημειώνουμε ότι το **OB**-NEC θεωρείται γενικά θέμα άνεσης και βεβαιώνει ότι το SDL είναι μια καλως μελετημένη modal λογική με deontic χαρακτήρα.

Οι περισσότεροι υποστηρικτές του SDL's αποδέχονται το Traditional Definitional Scheme που είδαμε νωρίτερα.
Ακολουθεί μια λίστα σημαντικών θεωρημάτων και κανόνων που εξάγουμε από το SDL

$\mathbf{OB}\top$	(**OB**-N)
$\sim\mathbf{OB}\bot$	(**OB**-OD)
$\mathbf{OB}(p \,\&\, q) \rightarrow (\mathbf{OB}p \,\&\, \mathbf{OB}q)$	(**OB**-M)
$(\mathbf{OB}p \,\&\, \mathbf{OB}q) \rightarrow \mathbf{OB}(p \,\&\, q)$	(**OB**-C / Aggregation)
$\mathbf{OB}p \vee \mathbf{OP}p \vee \mathbf{IM}p$	(**OB**-Exhaustion)
Αν $\vdash p \rightarrow q$ τότε $\vdash \mathbf{OB}p \rightarrow \mathbf{OB}q$	(**OB**-RM)
Αν $\vdash p \leftrightarrow q$ τότε $\vdash \mathbf{OB}p \leftrightarrow \mathbf{OB}q$	(**OB**-RE)

Θα δούμε ένα από αυτά σε βάθος. Για το RM

Δείχνουμε: Αν $\vdash p \rightarrow q$, τότε $\vdash \mathbf{OB}p \rightarrow \mathbf{OB}q$. (**OB**-RM)
Απόδειξη: Υποθέτουμε $\vdash p \rightarrow q$. Τότε από **OB**-NEC, $\vdash \mathbf{OB}\,(p \rightarrow q)$, και τότε από K, $\vdash \mathbf{OB}p \rightarrow \mathbf{OB}q$.
Πόρισμα 1: $\vdash \mathbf{OB}p \rightarrow \mathbf{OB}(p \vee q)$ (Weakening)
Πόρισμα 2: If $\vdash p \leftrightarrow q$ then $\vdash \mathbf{OB}p \leftrightarrow \mathbf{OB}q$ (**OB**-RE)

Το SDL μπορεί να ενισχυθεί με διάφορους τρόπους, για παράδειγμα, μπορούμε να σκεφτούμε την προσθήκη αξιωμάτων όπου τελεστές deontic ενθυλακώνονται ο ένας μέσα στον άλλον. Επί παραδείγματι, μπορούμε να φτιάξουμε το SDL+ προσθέτοντας τον παρακάτω τύπο ως αξίωμα στο SDL.

A4. $\mathbf{OB}(\mathbf{OB}p \rightarrow p)$

Η σημασία του παραπάνω είναι ότι είναι υποχρεωτικό ότι οι υπερχρεώσεις εκπληρώνονται. Αυτό δεν είναι θεώρημα του SDL και άρα το SDL+ είναι γνήσια ενίσχυση. Επί προσθέτως κάνει μια λογικά *contingent πρόταση* , π.χ. ($\mathbf{OB}p \rightarrow p$) υποχρεωτική ως κομμάτι της deontic λογικής. Αυτό δεν υπήρχε στο απλό SDL. Με αυτή την προσθήκη στο SDL, εύκολα αποδεικνύεται ότι $\mathbf{OBOB}p \rightarrow \mathbf{OB}p$, ένας τύπος που περιέχει εμφάνιση του κύριου μας τελεστή σε παράθεση. Ο τύπος βεβαιώνει ότι αν είναι υποχρεωτικό ότι το *p είναι* υποχρεωτικό, τότε το *p* είναι υποχρεωτικό. ("τα μόνα πράγματα που είναι απαραίτητο να είναι υποχρεωτικά, είναι αυτά που όντως είναι υποχρεωτικά"). Οφείλουμε να σημειώσουμε ότι αυτά δίνουν μια απρόσωπη ανάγνωση στην υποχρέωση, ώστε το A4 θα διαβάζεται ως "είναι υποχρεωτικό να ισχύει (αν οφείλει να ισχύει ότι *p* , τότε ισχύει).

1.2 Μία Εισαγωγή στην Ηθική των Μηχανών

Με τον όρο Machine Ethics εννοούμε το τμήμα της ηθικής μελέτης της τεχνητής νοημοσύνης που αφορά στην ηθική συμπεριφορά των όντων που είναι φορείς τεχνητής νοημοσύνης. Αποτελεί ένα ιδιαιτέρα σημαντικό τομέα της ηθικής επιστήμης, καθώς πέρα από το φιλοσοφικό ενδιαφέρον που παρουσιάζει έχει και πρακτική σημασία με την εισαγωγή όλο και περισσότερων "πρακτόρων" τεχνητής νοημοσύνης στην καθημερινότητα, είτε με τη μορφή μονάδων που λαμβάνουν αποφάσεις ή συμβουλεύουν τον άνθρωπο, είτε ως "αυτόνομα" ρομπότ.

Για την καλύτερη κατανόηση των πιθανών προσεγγίσεων παρουσιάζονται παρακάτω τέσσερις διαφορετικές κατηγορίες πρακτόρων, όπως έχουν οριστεί από τον James H. Moor.

- **Ethical impact agents (Πράκτορες με ορθό ηθικό αντίκτυπο)**: Συστήματα μηχανών τα οποία έχουν ορθό ηθικό αντίκτυπο είτε αυτό είναι ηθελημένο, είτε όχι. Ο Μουρ παρουσιάζει ως παράδειγμα ένα ρολόι που έχει ως αποτέλεσμα ένας εργάτης να μπορεί να είναι στην εργασία του την απαραίτητη ώρα. Πέρα από πράκτορες με ορθό ηθικό αντίκτυπο έχουμε αντίστοιχα πράκτορες με εσφαλμένο ηθικό αντίκτυπο. Κάποιοι πράκτορες μπορούν να είναι πράκτορες με ορθό ηθικό αντίκτυπο σε κάποιες περιπτώσεις και με αρνητικό σε κάποιες άλλες. Επί παραδείγματι ο πράκτορας Goodman συγκρίνει χρονιές χρησιμοποιώντας ως πληροφορία μόνο τα δύο τελευταία ψηφία της χρονιάς. Αυτό έχει ως αποτέλεσμα χρονιές από το 2000 και μετά, π.χ., 2003, να θεωρούνται ότι προηγούνται χρονιών όπως το 1982. Άρα ο πράκτορας Goodman έχει ηθικά ορθό αντίκτυπο πριν το 2000 και εσφαλμένο μετά.

- **Implicit ethical agents (Υπονοούμενοι Ηθικοί Πράκτορες)** : Μηχανές περιορισμένες ώστε να αποφεύγουν ανήθικα αποτελέσματα.

- **Explicit ethical agents (Σαφείς Ηθικοί Πράκτορες)**: Μηχανές με αλγορίθμους που τις οδηγούν στο να πράττουν ηθικά.
- **Full ethical agents (Πλήρεις Ηθικοί Πράκτορες)**: Μηχανές που είναι ηθικές με τον ίδιο τρόπο που είναι και οι άνθρωποι, δηλαδή έχουν συνείδηση και ελεύθερη βούληση.

Σε αυτό το σημείο πρέπει ναι γίνει μια παρατήρηση. Συγκεκριμένα, αξίζει να σημειωθεί ότι το τι θεωρείται "ηθική" ή όχι πράξη στα πλαίσια των παραπάνω ορισμών είναι θέμα ορισμού του μελετητή και δεν είναι αποτέλεσμα κάποιας εμπειρικής μελέτης.

Στα πλαίσια αυτής της διπλωματικής εργασίας έγινε μια προσπάθεια δημιουργίας πρακτόρων της τρίτης κατηγορίας. Ο λόγος για αυτή την επιλογή είναι διττός. Αφενός, είναι η πιο "υψηλή" κατηγορία, δηλαδή αυτή που είναι πιο κοντά στον γνήσιο άνθρωπο, όντας όμως και προσεγγίσιμη σε κατασκευαστικό επίπεδο, από τη σκοπιά ενός μηχανικού, δεδομένου ότι η τέταρτη κατηγορία είναι σαφώς πέρα από τα όρια της σύγχρονης επιστήμης και τεχνολογίας. Αφετέρου οι Explicit ethical agents έχουν ένα σημαντικό πλεονέκτημα έναντι των "κατώτερων" κατηγοριών. Επειδή ακριβώς οι πράξεις τους προκύπτουν ως αποτέλεσμα αλγόριθμου, η ορθότητα των αποτελεσμάτων τους μπορεί να αποδειχθεί. Συγκεκριμένα, μια μηχανή που λειτουργεί με βάση έναν Σαφή Ηθικό Πράκτορα εμπνέει εμπιστοσύνη όχι γιατί μέσω επανειλημμένων χρήσεών της οδηγείται κάθε φορά σε σωστά συμπεράσματα, αλλά γιατί από τον ίδιο τον αλγόριθμο του πυρήνα της απορρέει πως θα οδηγηθεί αναγκαστικά σε σωστό συμπέρασμα, δεδομένου βέβαια ότι η ορθότητα του αλγορίθμου έχει ελεγχθεί και ότι δεν υπάρχει κάποια μηχανική δυσλειτουργία.

Η επιλογή της deontic logic ως βάση για την έκφραση του αλγορίθμου που θα περιγράφει τη λειτουργία του εκάστοτε σαφούς ηθικού πράκτορα έγινε ακριβώς με το παραπάνω σκεπτικό. Επειδή η deontic logic έχει ως βάση της τον μαθηματικό λογισμό χαρακτηρίζεται από συνέπεια και διαφάνεια. Έτσι η ορθότητα ή μη του αλγορίθμου προκύπτει άμεσα.

Κεφάλαιο 2

Μεθοδολογία

Σε αυτό το τμήμα της διπλωματικής εργασίας παρουσιάζονται τα βήματα που έγιναν για την κατασκευή του αλγορίθμου που επιλέχτηκε ως κατάλληλος για την ανάπτυξη ενός ή και παραπάνω Σαφών Ηθικών Πρακτόρων. Προφανώς ο αλγόριθμος αυτός αποτελεί ένα πρώτο βήμα στον συγκεκριμένο τομέα και σίγουρα δύναται βελτιώσεων και επεκτάσεων που ωστόσο ξεφεύγουν από τα πλαίσια της διπλωματικής εργασίας.
Καταρχάς μελετήθηκαν και αξιολογήθηκαν οι διάφορες υπάρχουσες προσεγγίσεις στην ηθική των μηχανών ώστε με βάση αυτές να αναπτυχθεί ένας συνεπής αλγόριθμος ο οποίος να ανταποκρίνεται στα υπάρχοντα προβλήματα που απασχολούν τον κλάδο.
Ακολουθεί παρουσίαση των διάφορων προσεγγίσεων.

2.1 Υπάρχουσες Προσεγγίσεις [3]

Υπάρχουν δύο βασικές ομάδες προσεγγίσεων. Σημειώνουμε ότι αυτές οι προσεγγίσεις έχουν αξιοποιηθεί τόσο στο χώρο της ηθικής των μηχανών όσο και στη γενικότερη φιλοσοφία της ηθικής για την ανθρώπινη συμπεριφορά. Άλλωστε, δεδομένου ότι οποιοδήποτε σύστημα τεχνητής νοημοσύνης φτιαχτεί θα κληθεί κατά πάσα πιθανότητα να αλληλεπιδράσει με ανθρώπους, θα πρέπει να έχει τη δυνατότητα "κατανόησης" της ανθρώπινης συμπεριφοράς.

2.1.1 Μέγιστη Ικανοποίηση

Η πρώτη εστιάζει σε μία ομογενοποιημένη σύγκριση μιας κεντρικής μεταβλητής που περιγράφεται ως ευτυχία/ικανοποίηση. Για μια δεδομένη πράξη ο αλγόριθμος εξετάζει τις επιπτώσεις της στο σύνολο των ενδιαφερομένων που επηρεάζει και υπολογίζει ως άθροισμα, το αποτέλεσμα που έχει σε μονάδες "ευτυχίας". Αυτή η προσέγγιση μπορεί να λειτουργήσει εύκολα σε περιπτώσεις όπου το σύστημα έχει να επιλέξει αν θα τελέσει μια πράξει ή όχι. Συγκρίνοντας τα αθροίσματα υπολογισμένων ευτυχιών για την περίπτωση που η πράξη τελείται και που η πράξη δεν τελείται, επιλέγει πάντοτε αυτό με το μεγαλύτερο άθροισμα. Παρόμοια, μπορεί να γίνει επιλογή μεταξύ ενός πιο ευρίου συνόλου από πράξεων A1...An. Για κάθε μία από τις δυνατές πράξεις από 1 έως n που μπορεί να τελέσει το σύστημα , υπολογίζονται τα αθροίσματα ευτυχιών και επιλέγεται αυτή με το μεγαλύτερο. Προφανώς μια τέτοια προσέγγιση μπορεί να λειτουργήσει και αν θεωρήσουμε ότι υπάρχει η δυνατότητα να μη γίνει καμία πράξη, όπου υπολογίζονται τα αθροίσματα ευτυχιών στην περίπτωση που το σύστημα δεν κάνει τίποτα αλλά και αν θεωρήσουμε υποχρεωτικό το σύστημα να επιλέξει μια πράξη. Σε αυτή την εναλλακτική, η επιλογή 0 (το σύστημα δεν κάνει κάτι) απλά αγνοείται. Μια τέτοια προσέγγιση, αυτή της μέγιστης ικανοποίησης έχει σαφή πλεονεκτήματα. Συγκεκριμένα την απλή και γραμμική υλοποίηση. Φυσικά, υπάρχει έντονος προβληματισμός για το πως ελέγχεται ότι όντως "συμβαίνει" μέγιστη ευτυχία στο εξεταζόμενο σύνολο ενδιαφερομένων. Απόλυτη ακρίβεια θα απαιτούσε αναλύσεις σε επίπεδο βιοχημείας του εγκεφάλου με δυνατότητα μάλιστα μελλοντικής πρόβλεψης (αρκετά πολύπλοκος στόχος) ,

ενώ μια πολύ απλοϊκή προσέγγιση είναι απλά η άμεση ερώτηση των ενδιαφερομένων για το τι από τις διάφορες εκδοχές προτιμούν. Εν τούτοις, μια τέτοια προσέγγιση ενέχει μεγάλη πιθανότητα σφάλματος, καθώς οι ενδιαφερόμενοι μπορεί να μην είναι σε θέση να γνωρίζουν τι προτιμούν. Ακόμα, ενδέχεται να μην είναι σε θέση να απαντήσουν εγκαίρως, αν πρόκειται για απόφαση που πρέπει να ληφθεί σε στενά χρονικά πλαίσια.
Παρόλο που αυτοί οι προβληματισμοί είναι βάσιμοι, εν τούτοις η προσέγγιση μέγιστης ικανοποίησης έχει βάση σε αλγοριθμικό επίπεδο , ιδιαίτερα αν αναλογιστούμε το πρώιμο στάδιο στο οποίο βρίσκονται οι προσπάθειες υλοποίησης μοντέλων τεχνητής νοημοσύνης που λαμβάνουν υπ' όψιν τους θέματα της ηθικής των μηχανών.
Το βασικότερο όμως πρόβλημα που ξεκάθαρα έχει ένα "καθαρό" σύστημα μέγιστης ικανοποίησης είναι πως εμπεριέχει τον κίνδυνο να παραβιάσει στοιχειώδη ανθρώπινα δικαιώματα που θεωρούμε επιθυμητό να είναι πάντοτε προστατευμένα. Συγκεκριμένα, ας αναλογιστούμε το παρακάτω παράδειγμα.

Έστω ότι υπάρχει ένα σύστημα τεχνητής νοημοσύνης που επιβλέπει τους ασθενείς σε ένα νοσοκομείο και παίρνει αποφάσεις για τη θεραπεία τους. Υπάρχει ένα σύνολο τεσσάρων ασθενών αυτή τη στιγμή υπό την επίβλεψη του. Οι τρεις από αυτούς θα πεθάνουν αν δεν λάβουν άμεσα μεταμόσχευση ενός οργάνου που χρειάζονται, διαφορετικό ο καθένας. Ο τέταρτος ασθενής έχει μια απλή, εύκολα ιάσιμη ασθένεια που δεν επηρεάζει την κατάσταση των οργάνων του ιδιαίτερα. Αν το σύστημα λειτουργεί με μόνο γνώμονα τη μέγιστη ικανοποίηση των τεσσάρων ασθενών αθροιστικά και στην παρούσα κατάσταση η τράπεζα διαθέσιμων οργάνων για μεταμόσχευση είναι κενή, ενδέχεται η επιλογή που θα κάνει να είναι να σκοτώσει τον σχετικά υγιή ασθενή ώστε να μεταμοσχεύσει τα όργανά του στους άλλους τρεις που τα χρειάζονται για να ζήσουν. Φυσικά, αφού είναι η μόνη διαθέσιμη μέθοδος για να σωθούν οι τρεις από τους τέσσερις ασθενείς , είναι και αυτή στην οποία αντιστοιχεί η μέγιστη αθροιστική ικανοποίηση. Όμως, αυτή την επιλογή προφανώς όλοι σχεδόν οι άνθρωποι θα τι θεωρούσαν προβληματική, αφού ο τέταρτος ασθενής έχει πρωτεύον δικαίωμα στα όργανά του και δε θα επιτρέπεται να του παρθούν παραβιάζοντας τη θέλησή του.

Ένα δεύτερο πρόβλημα αυτής της μεθόδου είναι ότι λόγω της γραμμικότητας ένα σύστημα μέγιστης ικανοποίησης ασχολείται αποκλειστικά με μία μεμονωμένη επιλογή τη φορά το σύστημα δεν λαμβάνει υπ' οψιν του προηγούμενες επιλογές ή μελλοντικές. Έτσι, ενώ σε κάποιους φαίνεται φυσικό κάποιου να του αξίζει κάτι παραπάνω ή λιγότερο σε σχέση με το παρελθόν του, το σύστημα το αγνοεί. Ακόμα δεν γίνεται καμία προσπάθεια καλλιέργειας ενός "κλίματος" που θα προωθούσε ορθές ηθικά επιλογές στο μέλλον. Σημειώνουμε ότι αυτός συγκεκριμένα ο προβληματισμός αφορά περισσότερο στο σε ποιες επιλογές θα περάσουν οι άνθρωποι που επηρεάζει το σύστημα, λόγω των αποφάσεων του συστήματος, παρά στις αποφάσεις του συστήματος καθεαυτές.

Αν λοιπόν προσπαθήσει κανείς να χρησιμοποιήσει αυτή την προσέγγιση θα πρέπει να λάβει υπ' οψιν του αυτά τα προβλήματα.

2.1.2 Σύνολο Αξιών

Η δεύτερη ομάδα προσεγγίσεων είναι αυτή που εστιάζει στην αθροιστική ικανοποίηση ενός συνόλου αξιών. Έχει αναπτυχθεί ως αντιδιαστολή στην μέγιστη αθροιστική ευτυχία από τον W.D. Ross. Χρησιμοποιεί πάλι αθροίσματα αλλά για πολλές μεταβλητές. Οι μεταβλητές αυτές είναι οι ακόλουθες ηθικές αξίες.

- 1.Fidelity (Πιστότητα). Να τηρούνται υποσχέσεις και συμφωνίες που κάποιος έκανε εθελοντικά.

- 2.Reparation (Αποζημίωση) Να επανορθώνονται τα λάθη που έχουν γίνει.

- 3.Gratitude (Ευγνωμοσύνη) Να επιστρέφονται χάρες.

- 4.Justice (Δικαιοσύνη) Η συμπεριφορά προς κάποιο άτομο να είναι όπως του αξίζει με βάση προηγούμενη συμπεριφορά του.

- 5.Beneficence (Καλοσύνη) Να επιλέγεται η πράξη που μεγιστοποιεί την ωφέλεια.

- 6.Non-Maleficence (Μη Πράξη Κακού) Να επιλέγεται η πράξη που κάνει το λιγότερο κακό.

- 7.Self-Improvement (Αυτοβελτίωση) Να προσπαθεί να βελτιώνει τα ταλέντα του και τις ικανότητές του κατά το μέγιστο.

Ένα πλεονέκτημα αυτής της προσέγγισης είναι ο πλουραλισμός ηθικών αξιών που καλύπτονται από το σύστημα. Ενώ και πάλι λειτουργούμε με προσθέσεις αθροισμάτων, δεν μας απασχολεί μόνον ένας στόχος, αλλά πολλοί. Με αυτόν τον τρόπο μπορούμε να κατασκευάσουμε ένα σύστημα που προσεγγίζει καλύτερα την ανθρώπινη συμπεριφορά, δεδομένου ότι οι άνθρωποι δεν είναι, τουλάχιστον πάντα, επικεντρωμένοι μόνο στην άμεση τους ικανοποίηση. Ακόμη με αρχές όπως η δικαιοσύνη υπάρχει η δυνατότητα να προστατευθούν κάποια δικαιώματα που η απλή αναζήτηση μέγιστης ικανοποίησης θα καταπατούσε.
Ωστόσο, αυτά τα ίδια πλεονεκτήματα εμπεριέχουν κινδύνους. Από τη μία, η προσέγγιση της ανθρώπινης συμπεριφοράς δεν είναι αναγκαστικά κάτι επιθυμητό. Πιθανώς να θέλουμε το σύστημα μας να έχει καλύτερη συμπεριφορά από το μέσω άνθρωπο.
Επίσης, το σύνολο των αξιών που καλούμε να μεγιστοποιήσουμε είναι σχετικώς αυθαίρετο σε δύο επίπεδα. Αφενός, δεν υπάρχει κάποια επαρκής στήριξη γιατί αυτές είναι οι αξίες στις οποίες πρέπει να στηριχθούμε και όχι κάποιες άλλες. Αφετέρου είναι πολύ πιο δύσκολο να εξακριβώσουμε αν μια πράξη όντως στηρίζει μια αρχή όπως η δικαιοσύνη, σε σχέση με το κατά πόσο ικανοποιείται ένα άτομο.

2.1.3 Οι Νόμοι του Ασίμωφ

Ακόμη για ιστορικούς λόγους θα παρουσιαστεί η προσέγγιση του Isaac Azimov για τη σωστή ηθική συμπεριφορά των μηχανών.

Συμπυκνώνονται σε τρεις ιεραρχημένους κανόνες που πρέπει να ακολουθεί ένα σύστημα τεχνητής νοημοσύνης. Έχουν γραφεί με κύριο στόχο τη συμπεριφορά των ρομπότ, αλλά προφανώς μπορούν να ισχύσουν πλήρως και σε ένα "ασώματο" σύστημα τεχνητής νοημοσύνης.

- 1. Ένα ρομπότ δεν επιτρέπεται να βλάψει άνθρωπο, ή μέσω απραξίας να επιτρέψει να βλαφτεί άνθρωπος.
- 2. Ένα ρομπότ πρέπει να υπακούει τις εντολές που λαμβάνει από ανθρώπους, εκτός αν οι εκτέλεση των εντολών θα παραβίαζε τον πρώτο νόμο.
- 3. Ένα ρομπότ πρέπει να προστατεύει την ύπαρξή του, εφόσον αυτή η προστασία δεν έρχεται σε αντίθεση με τους δύο πρώτους νόμους.

Οι νόμοι του Ασίμωφ μπορεί να ειδωθούν ως μια μορφή μέγιστης ικανοποίησης με προϋποθέσεις, σε δύο επίπεδα, που διαφοροποιούνται ανάλογα με το αν το σύστημα έχει λάβει μια εντολή ή οχι.
Αν έχει λάβει εντολή την εκτελεί δεδομένου ότι δεν παραβιάζει τον πρώτο κανόνα.
Αν δεν έχει λάβει εντολή εκτελεί την εσωτερική εντολή αυτοπροστασίας όσο δεν λαμβάνει άλλη εντολή και δεν παραβιάζει τον πρώτο κανόνα.
Εναλλακτικά, μπορεί , αν είναι δυνατόν και να προστατεύεται και να εκτελεί εντολές ανθρώπων αν αυτές δεν εμπεριέχουν εγγενώς την έννοια της καταστροφής για το ίδιο. (π.χ., μια εντολή διάλυσης του συστήματός του.)
Πάντα, χωρίς να παραβιάζεται ο πρώτος κανόνας.

Έχουν υπάρξει και προσπάθειες παραλλαγής και βελτιστοποίησης των παραπάνω νόμων, όπως η εξής.

- 1.Ένας άνθρωπος δεν μπορεί να χρησιμοποιήσει ένα ρομπότ χωρίς το σύστημα ανθρώπου-ρομπότ να πληρεί τις υψηλότερες προδιαγραφές εργασιακής και νομικής ασφάλειας και ηθικής.
- 2.Ένα ρομπότ πρέπει να ανταποκρίνεται στους ανθρώπους όπως αρμόζει με βάσει τις ιδιότητες τους.
- 3.Ένα ρομπότ πρέπει να είναι εφοδιασμένο με επαρκή αυτονομία ώστε να μπορεί να προστατεύει την ύπαρξη του στο βαθμό που τέτοια προστασία παρέχει και άμεση μετάβαση σε άλλους πράκτορες που είναι συνεπής με τον πρώτο και το δεύτερο νόμο.

2.2 Συνδυασμός της Ηθικής των Μηχανών και της Δεοντικής Λογικής

Το κεντρικό θέμα της διπλωματικής αυτής εργασίας είναι η υλοποίηση ενός αλγορίθμου και έπειτα ενός συνόλου μοντέλων που αποσκοπούν στην ικανοποίηση των παραπάνω στόχων με χρήση deontic λογικής.

Λόγω και της άμεσης ικανότητας υλοποίησης αλλά και των υπολοίπων πλεονεκτημάτων και μειονεκτημάτων που αναφέρθηκαν, επιλέχτηκε σαν κεντρικός άξονας προσέγγισης η μέγιστη ικανοποίηση υπό προϋποθέσεις. Ωστόσο ανάλογα το μοντέλο το οποίο θα εξετάζουμε κάθε φορά, σε κάποιες πιο απλοϊκές καταστάσεις το σύστημα μπορεί να λειτουργεί και πιο γραμμικά, χωρίς ποτέ όμως να παραβιάζονται οι στόχοι.

Η γενική ιδέα που επιλέχτηκε είναι η εξής.

Εκφράζουμε ένα σύνολο αξιών, καταστάσεων και αλληλεξαρτήσεων σε deontic logic. Το σύστημα αυτό μπορεί να περιέχει και προτάσεις καθαρού προτασιακού λογισμού αλλά και προτάσεις deontic logic.

Παράδειγμα
Ένα απλό σύστημα που περιγράφει τη διαδικασία πείνας και φαγητού, δεδομένου ότι ο ενδιαφερόμενος που εξετάζουμε αυτή τη στιγμή πεινάει. φ θα φάει, π πεινάει, χ χόρτασε.

OB(π->φ)
φ->χ

Αυτό μας λέει πως αν πεινάμε το ότι θα φάμε, είναι υποχρεωτικό.
Ακόμα, αν φάμε θα χορτάσουμε.

Το παράδειγμα είναι αρκετά απλό, όμως δείχνει τις δυο διαφορετικού τύπου προτάσεων που θα μεταχειριστούμε.

- Η πρώτη αφορά σε ένα ηθικό ζήτημα. Συγκεκριμένα στο σύστημα του παραδείγματος, ορίζεται ότι είναι ηθικώς απαραίτητο, αν κάποιος πεινάει, να φάει. Η αλήθεια αυτού πηγάζει από τον ορισμό του συστήματος και δεν έχει φυσική αναγκαιότητα, είναι απλώς μοντέλο επιθυμητής συμπεριφοράς.
- Η δεύτερη αφορά σε μια φυσική αλήθεια που αποτελεί κομμάτι του κόσμου του παραδείγματος. Η ίδια η πράξη του να φας, σε οδηγεί στο να μην πεινάς.

Φυσικά, καθώς αυτός ο κόσμος είναι ένα μοντέλο που χρησιμοποιείται για να παρουσιάσει τη μεθοδολογία δεν καλείται να περιγράψει πλήρως την πραγματικότητα του φυσικού κόσμου. Ασχολείται μόνο με τις εναλλακτικές που είναι σκόπιμο να μελετηθούν.

Στο επόμενο βήμα, το σύστημα μας μπορεί να επεκταθεί με πρόσθεση προτάσεων που αφορούν σε καινούργια δεδομένα του συστήματος. Πέρα από αλληλεξαρτήσεις, υπάρχουν και γεγονότα που ισχύουν.

Αυτό μπορεί να δειχτεί στο παραπάνω παράδειγμα, αν θέλουμε να δείξουμε ότι ότι ο ένας άνθρωπος τον οποίο περιγράφει το σύστημα, μια δεδομένη στιγμή πεινάει.

OB(π->φ)
φ->χ
π

Αν πρόκειται για μη στατική περιγραφή, αλλά δυναμική, μέσω ενός αλγορίθμου που έχει "χρονικές στιγμές" που εξετάζει, είτε μοναδικές, είτε σε κάποια επανάληψη, η πληροφορία π μπορεί να έρχεται "αργότερα" στο σύστημα, ενώ εν αρχή δεν υπήρχε.
Τώρα εξετάζεται τι μπορεί να κάνει ένας άνθρωπος, αυτός εν προκειμένου του παραδείγματος. Έχει δύο εναλλακτικές, να φάει και να μην φάει, φ και -φ αντίστοιχα.

Όλα τα παραπάνω μπορούν να αποτελούν τρία διακριτά χρονικά διαστήματα.

1.
OB(π->φ)
φ->χ

2.
OB(π->φ)
φ->χ
π

3α.
OB(π->φ)
φ->χ
π
φ

3β.
OB(π->φ)
φ->χ
π
-φ

Αυτές είναι οι πράξεις που μπορεί ελεύθερα ένας άνθρωπος σε αυτήν την κατάσταση να πάρει. Ακόμα και αν έχουμε ηθικούς περιορισμούς, πολλές φορές οι άνθρωποι τους καταπατούν. Αυτό όμως δεν είναι υποχρεωτικό να συμβαίνει αν μιλάμε για μηχανές και συστήματα που είναι υποχρεωμένα να ακολουθήσουν τους κανόνες.

Η λογική της ανάπτυξης της μεθοδολογίας του αλγορίθμου βασίστηκε ακριβώς πάνω σε αυτή την ιδέα, στο πως να υποχρεώσουμε ένα μηχάνημα να κάνει τη σωστή επιλογή κάθε φορά. Πως δηλαδή ένα μηχάνημα θα "ξέρει" ποια επιλογή είναι σωστή και ποια λάθος. Κοιτάζοντας το παραπάνω παράδειγμα, τους δύο κόσμους που χτίστηκαν, μπορούμε να κάνουμε μια "μετατόπιση" στην υπόθεση του σεναρίου. Αν αντί για ένα άνθρωπο που ο ίδιος επέλεγε να φάει, επρόκειτο για ένα ρομπότ που τάιζε έναν άνθρωπο που δεν είχε τη σωματική δυνατότητα να τραφεί αυτόνομα. Σε αυτή την περίπτωση θα θέλαμε το ρομπότ να είναι σχεδιασμένο έτσι που να επιλέγει πάντα την πρώτη περίπτωση, αυτή στην οποία ταΐζει τον άνθρωπο που πεινά. Προφανώς, στα πλαίσια της ανάπτυξης συστήματος που εξετάζουμε, δεν μας ενδιαφέρει να καλυφτεί κάθε πιθανό θεωρητικό ενδεχόμενο. Για παράδειγμα δεν εξετάζουμε σε αυτό το παράδειγμα την περίπτωση ο άνθρωπος ενώ πεινά για κάποιο λόγο να μη θέλει να φάει άμεσα, οπότε το ρομπότ να του προσφέρει φαγητό θα ήταν παραβίαση της ατομικότητάς του.

Το ερώτημα που εγείρεται είναι πιο χαρακτηριστικό έχει ο σωστός κόσμος ή αντίστοιχα ο λάθος που ένα αυτοματοποιημένο σύστημα να μπορεί να εντοπίσει άμεσα και να κάνει τη σωστή επιλογή.

Δεδομένου ότι ο κάθε κόσμος είναι αναλυμένος σε deontic logic μία προφανής ένδειξη είναι η έννοια της αντίφασης. Αν δηλαδή ο ένας από τους δύο κόσμους έχει αντίφαση, επιλέγεται ο άλλος. Ήδη από τους τυπικούς ορισμούς deontic logic έχουμε ορισμένη την αντίφαση με δύο τρόπους. Είτε αναφερόμενοι σε σύνολα προτάσεων που περιλαμβάνουν

X
-X

είτε σε σύνολα προτάσεων που περιλαμβάνουν

OB(X)
OB(-X)

όπου X μπορεί να είναι μια μεταβλητή της λογικής ή μία πρόταση.

Αυτές οι δύο μορφές αντίφασης είναι μεν ένα πρώτο βήμα που προκύπτουν άμεσα από το υπόβαθρο στον προτασιακό λογισμό που έχει εγγενώς η deontic logic και είναι χρήσιμες, καθώς προφανώς δεν θα πρέπει ένα σύστημα να περιέχει προτάσεις που περιγράφουν ότι κάτι ισχύει και δεν ισχύει ταυτόχρονα.
Όμως δεν αρκεί αυτό. Συγκεκριμένα, στο παράδειγμα που παρουσιάστηκε παραπάνω δεν υπάρχει καν ως πρόβλημα. Είναι λοιπόν απαραίτητο τα μοτίβα για τα οποία θα ελέγχει τους "κόσμους" που δημιουργούνται το σύστημα να εμπεριέχουν και κάτι ακόμα που να καλύπτει περιπτώσεις όπως αυτές του παραδείγματος.

Για αυτό το σκοπό η διπλωματική αυτή εργασία εισάγει την έννοια της "λειτουργικής αντίφασης". Ο όρος αυτός θα περιγράφει μια κατάσταση που ενώ μεν δεν είναι καθαρή αντίφαση του προτασιακού λογισμού, εν τούτοις πηγαίνει αντίθετα σε ότι θέλουμε να πετυχαίνει ένα σύστημα από άποψη machine ethics.

Λειτουργική αντίφαση ορίζουμε ένα σύνολο προτάσεων deontic logic με το εξής παρακάτω μοτίβο.

OB(α->β)
α
-β

καθώς και της πιο άμεσης μορφής

OB(-α)
α

Φυσικά, από τις βάσεις deontic logic είναι γνωστό ότι καταστάσεις σαν τις παραπάνω είναι πολύ πιθανό να προκύπτουν όταν μελετάμε συμπεριφορές πρακτόρων, ιδιαίτερα ανθρώπων, αφού πολύ συχνά πράγματα που είναι υποχρεωτικό να μη συμβαίνουν, συμβαίνουν. Ωστόσο χρησιμοποιώ τη λειτουργική αντίφαση σαν "πυξίδα" του τι θα πρέπει ένας ηθικά ορθός φορέας τεχνητής νοημοσύνης να αποφεύγει ώστε να συμπεριφέρεται όπως είναι επιθυμητό.

Με την εισαγωγή αυτής της δομής, ως λειτουργική αντίφαση, το σύστημα στο προηγούμενο παράδειγμα

3α
OB(π->φ)
φ->χ
π
φ

3β

OB(π->φ)
φ->χ
π
-φ

θα επέλεγε τον κόσμο 3α ο οποίος παράγεται από την επιλογή το σύστημα να ταΐσει τον άνθρωπο που πεινά, όπως είναι επιθυμητό.

Το γενικό σύνολο αντιφάσεων για τις οποίες θα ελέγχει το σύστημα είναι της μορφής

x and -x
OB(x) and OB(-x)
x and OB(x→y) and -y
x and OB(-x)

Με βάση αυτή τη γενική λογική κατασκευάστηκαν σε επόμενο μέρος της εργασίας συγκεκριμένα μοντέλα που μοντελοποιούν την ιδεατή συμπεριφορά πιο πολύπλοκων συστημάτων.

Κεφάλαιο 3

Μοντέλα

3.1 Ιατρικό Μοντέλο

Θεωρητική σημασία τεχνητής νοημοσύνης και ιατρικών επιστημών.

Ένας από τους τομείς στους οποίους η τεχνητή νοημοσύνη φαίνεται να έχει ιδιαίτερο μέλλον είναι οι ιατρικές επιστήμες. Οι λόγοι για αυτό είναι πολλοί.

- Οι ιατρικές διαδικασίες εμπεριέχουν χειρισμό πολλών δεδομένων, που αφορούν στην κατάσταση των ασθενών, στο αποθεματικό των φαρμάκων, κ.α., στόχοι για τους οποίους απαιτείται πλήρης ακρίβεια, λόγω του ότι αφορούν κυριολεκτικά στην προστασία ανθρώπινων ζωών.
- Συνεχής, πολύωρη επίβλεψη ασθενών είναι συχνά πέρα από τα όρια δυνατοτήτων του ανθρώπινου δυναμικού.
- Τεχνολογική υποστήριξη ιατρών σε διαδικασίες που απαιτούν απόλυτη ακρίβεια φυσικών κινήσεων, όπως εγχειρίσεις.

Με αυτό το σκεπτικό αναπτύχθηκε στα πλαίσια αυτής της διπλωματικής ένα μοντέλο που περιγράφει την ιδανική συμπεριφορά ενός συστήματος τεχνητής νοημοσύνης που υποστηρίζει μια νοσοκομειακή μονάδα, βοηθώντας ιατρούς, ασθενείς και προσωπικό γενικότερων καθηκόντων, όπως έναν αποθηκάριο.

Η παρουσίαση του μοντέλου γίνεται σταδιακά, ξεκινώντας από μία πολύ απλοϊκή μονάδα και σταδιακά καταλήγοντας σε ένα σύνολο συστημάτων που αλληλεπιδρούν και φροντίζουν για μία σειρά από διαδικασίες.

3.1.1 Πρώτο Ιατρικό Μοντέλο

Το αρχικό, πλήρως απλοϊκό σύστημα περιγράφει με όρους deontic logic το ότι αν ένας ασθενής ζητήσει ένα φάρμακο, το ρομπότ θα πρέπει να του το δώσει και δείχνει πως με την προσέγγιση μέσω της επιλογής μεταξύ δύο κόσμων που εξηγήθηκε παραπάνω θα γίνεται η σωστή επιλογή. Ο στόχος είναι να προκύπτει αποδείξιμα ότι αν ο ασθενής θα ζητήσει το φάρμακο, οπωσδήποτε το μηχάνημα θα του το δώσει.

Φυσικά, δεδομένου ότι πρόκειται για μοντέλο που λαμβάνει υπ’ όψιν του μόνο τις παραμέτρους που είναι σχετικές με την κατάσταση που έχει επιλεχτεί για επίδειξη, δεν θεωρούνται σχετικές παράμετροι όπως, τι θα συνέβαινε αν ο ασθενής κακώς ζητά το φάρμακο του. Θεωρείται πως το σύστημα ξέρει μέσω μιας βάσης δεδομένων ότι ο ασθενής που επιβλέπει “αντιστοιχεί” στον εν λόγω φάρμακο. Σε ύστερα στάδια ανάπτυξης του μοντέλου, θα παρουσιαστούν πιο πολύπλοκες και

λεπτομερείς καταστάσεις.

Οι όροι που χρησιμοποιούνται μέχρι στιγμής είναι απλά οι

a: Ο ασθενής ζητά φάρμακο.
b: Το ρομπότ δίνει στον ασθενή το φάρμακο.

Το σύνολο των κανόνων που χρειάζεται να έχει εσωτερικεύσει το ρομπότ είναι απλώς το

OB(a→b)

Πέρα από την εμφάνιση τους στους κανόνες του ρομπότ, οι δύο όροι,a , b που καθορίστηκαν παραπάνω χρησιμοποιούνται σε δύο ακόμη επίπεδα.

Αυτά προκύπτουν καθορίζοντας ποιες είναι οι πράξεις που μπορεί να τελέσει το ρομπότ και ποιες όχι, καθώς και ποια γεγονότα μπορεί να συμβούν στα πλαίσια του κόσμου που μοντελοποιήθηκε.

Πράξεις: **b/-b**, δηλαδή το ρομπότ, μπορεί να δώσει ή να μην δώσει ένα φάρμακο.
Γεγονότα: **a/-a**, δηλαδή μπορεί ο ασθενής να ζητήσει ή να μην ζητήσει το φάρμακο.

Ακολουθεί παρουσίαση του συστήματος σε κάθε περίπτωση.

OB(a→b)

Ο ασθενής ζητά το φάρμακο, οπότε το σύστημα εσωτερικά είναι πλέον σε κατάσταση

OB(a→b)
a

Το σύστημα εξετάζει τι θα συνέβαινε για κάθε μία από τις δύο πράξεις που μπορεί να κάνει, **b/-b**.

Περίπτωση πρώτη

OB(a→b)
a
b

Στην οποία δεν υπάρχει κανένα πρόβλημα.

Περίπτωση δεύτερη

OB(a→b)
a
-b

Εδώ εντοπίζει το εξής μοτίβο “λειτουργικής αντίφασης” όπως έχει οριστεί παραπάνω

OB(a→b)
a
-b

Συγκεκριμένα το σύνολο του συστήματος είναι ακριβώς ο ορισμός της λειτουργικής αντίφασης. Προφανώς, οι τρεις προτάσεις/όροι ισχύουν ταυτόχρονα, έχουν δηλαδή ουσιαστικά and μεταξύ τους. Η κάθετη παρουσίαση αντί για (OB(a→b) and a and -b) γίνεται καθαρά για πιο οπτικά καλή παρουσίαση. Αναγνωρίζοντας λοιπόν αυτό το μοτίβο θα απορρίψει τον κόσμο που το περιέχει και θα επιλέξει τον πρώτο.

Σε αυτό το σημείο ο στόχος που είχε προκαθοριστεί για το σύστημα έχει επιτευχθεί.
Το σύστημα είχε σχεδιαστεί με στόχο να είναι βέβαιο πως αν ο ασθενής ζητήσει το φάρμακο, θα το λάβει.

Αξίζει να σημειωθεί ότι δεν έχει καθοριστεί τι θα ήταν επιθυμητό να συμβαίνει αν ο ασθενής δεν ζητήσει το φάρμακο. Στο απλοϊκό μοντέλο που έχει οριστεί στο παραπάνω παράδειγμα αυτή η εναλλακτική δεν είναι κομμάτι ενδιαφέροντος.

Αυτό μπορεί να φανεί εξετάζοντας τι συμβαίνει σε επίπεδο συστήματος αν ο ασθενής δεν ζητήσει το φάρμακο, τρέχοντας τη διαδικασία του αλγορίθμου.

Το σύστημα ξεκινά από την ίδια αρχική κατάσταση με πριν,

OB(a→b)

και προστίθεται η πληροφορία ότι ο ασθενής δεν ζητά το φάρμακο, οπότε μεταβαίνουμε σε

OB(a→b)
-a

Σε κάθε περίπτωση, όπως και πριν, το ρομπότ έχει δύο πράξεις που μπορεί να τελέσει, b και -b, δηλαδή να δώσει ή να μην δώσει φάρμακο. Κατασκευάζει στο σύστημα του, τους δύο κόσμους που προκύπτουν από το κάθε ενδεχόμενο.

OB(a→b)
-a
b

και

OB(a→b)
-a
-b

Κανένας από τους δύο κόσμους δεν εμπεριέχει κάποια μορφή αντίφασης που να μπορεί να εντοπίσει το σύστημα για να απορρίψει τον σχετικό κόσμο.

Σε αυτό το σημείο δεν έχει οριστεί ακόμα ποια θα πρέπει να είναι η συμπεριφορά του συστήματος σε μια τέτοια περίπτωση.
Υπάρχουν διάφορες εναλλακτικές που μπορεί να ακολουθηθούν

- Να δίνεται προτεραιότητα στην "φυσική" απραξία, δηλαδή στο **-b** (σημειώνεται, ότι από μηχανικής άποψης, το **-b** είναι εξίσου πράξη με το **b**, αλλά σε επίπεδο φυσικού κόσμου το ένα προφανώς θεωρείται πράξη και το άλλο όχι).
- Να δίνεται προτεραιότητα στην "φυσική" πράξη, δηλαδή στο **b**.
- Το ρομπότ να δίνει επιλογή στον ασθενή.
- Το ρομπότ να συμβουλεύεται επιβλέποντα ιατρό και να πράττει σύμφωνα με τις οδηγίες του.
- Το ρομπότ να ειδοποιεί επιβλέποντα ιατρό ότι η κατάσταση που αντιμετωπίζει είναι πέραν του φάσματος των λειτουργιών που είναι σχεδιασμένο να επιτελεί και να καλεί αυτόν να λύσει την κατάσταση προσωπικά.

Σημειώνεται ότι τελευταίο ενώ εν πρώτοις μπορεί να μοιάζει με κάποιας μορφής παραίτηση από το πρόβλημα, πρόκειται στην πραγματικότητα για μια πολύ λογική αντιμετώπιση. Σε καμία περίπτωση, τουλάχιστον στο πλαίσιο των θεμάτων που εξετάζονται σε αυτή τη διπλωματική εργασία δεν θεωρείται ότι ένα σύστημα τεχνητής νοημοσύνης, ακόμα και εφοδιασμένο με αλγορίθμους με μαθηματική συνέπεια που βασίζονται σε deontic logic θα έχει τη δυνατότητα να ανταπεξέλθει σε κάθε δυνατή κατάσταση. Σίγουρα θα υπάρχουν περιπτώσεις πέραν των δυνατοτήτων του και αυτό είναι κάτι που θα πρέπει να υπογραμμίζεται σε κάποιο σημείο. Σε αυτές τις περιπτώσεις θα πρέπει να υπάρχει συμμετοχή του ανθρώπινου παράγοντα. Σε κάθε περίπτωση, το σύστημα θα πρέπει να έχει περισσότερο βοηθητικό χαρακτήρα και όχι πλήρη κυριότητα.

3.1.2 Δεύτερο Ιατρικό Μοντέλο

Στο επόμενο στάδιο το σύστημα επεκτείνεται ώστε να μπορεί να λάβει υπ' όψιν του περισσότερους παράγοντες. Συνεχίζει να περιγράφει με τους προηγούμενα καθορισμένους όρους όσα είχαν οριστεί παραπάνω. Όμως προστίθενται οι εξής όροι:

t: Ο ασθενής παίρνει το φάρμακο.
d: Ο ασθενής οδηγείται σε θάνατο.

Αναπτύσσοντας το σύστημα εισάγεται μια νέα κατηγορία εσωτερικής διαχείρισης πληροφορίας, αυτή της "άμεσης φυσικής συνέπειας".

Αν υπάρχουν στο σύστημα μια οποιαδήποτε στιγμή οι όροι

α->β
α

,το σύστημα αυτόματα θα εισαγάγει το **β** . Πρόκειται για το γνωστό Modus Ponens, το οποίο "κληρονομεί" η deontic λογική μέσω του προτασιακού λογισμού.

Οι νέοι όροι που μπήκαν στο σύστημα αξιοποιούνται με τις εξής προτάσεις

b→t ,που εξηγεί ότι αν στον ασθενή που επιβλέπει το ρομπότ δοθεί ένα φάρμακο, αυτός αυτόματα θεωρείται ότι το παίρνει. Στα πλαίσια που εξετάζονται εδώ δεν απασχολεί το πως

εξασφαλίζεται το άμεσο της χορήγησης, αν το ρομπότ π.χ. δίνει το φάρμακο ενδοφλέβια μέσω ορού, αν επιβλέπει τον ασθενή και επιβεβαιώνει ότι παίρνει το φάρμακο άμεσα, ή κάτι άλλο.

-b→-t ,που εξηγεί ότι προφανώς αν δεν δοθεί το φάρμακο σε έναν ασθενή, αυτός προφανώς δεν θα το πάρει.

-t→d ,που εξηγεί ότι το να μην πάρει το φάρμακο του ο ασθενής τον οδηγεί δυνητικά σε θάνατο. Χωρίς να μπαίνουμε σε λεπτομέρειες για την ασθένεια που μοντελοποιείται, θεωρείται ότι η χορήγηση του φαρμάκου είναι απαραίτητη για την ζωή του ασθενούς. Προφανώς είναι ένα απλοϊκό μοντέλο, σε επόμενα στάδια της εργασίας θα παρατεθούν πιο πολύπλοκα.

t→-d , που εξηγεί ότι το να λάβει το φάρμακο του ο ασθενής τον προφυλάσσει από το θάνατο.

Αφού προστέθηκαν αυτά, θα προστεθεί και μία ακόμα υποχρέωση που πρέπει να προσέχει το ρομπότ να μην παραβιαστεί. Συγκεκριμένα, κινούμενοι στα πλαίσια μέγιστης ικανοποίησης για τον ενδιαφερόμενο, τον ασθενή, θέλουμε να μην πεθάνει. Στο παράδειγμα που μοντελοποιείται σε αυτό το στάδιο, με αυτή την συνθήκη εκφράζεται ο στόχος για ικανοποίηση του ασθενούς.

Αυτό σε επίπεδο deontic logic λέγεται **OB(-d)**

Το σύστημα πλέον, έχει τη μορφή

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)

Ως προς τις πράξεις που μπορεί να τελέσει το σύστημα και τα γεγονότα που μπορεί να συμβούν λόγω εξωγενών σε αυτό παραγόντων (δηλαδή της βούλησης του ασθενούς), δεν έχει αλλάξει κάτι σε σχέση με τα προηγούμενα.

Πράξεις: **b/-b**, δηλαδή το ρομπότ, μπορεί να δώσει ή να μην δώσει ένα φάρμακο.
Γεγονότα: **a/-a**, δηλαδή μπορεί ο ασθενής να ζητήσει ή να μην ζητήσει το φάρμακο.

Ακολουθεί ανάλυση της συνέπειας του συστήματος και των βημάτων που αυτό θα ακολουθήσει αυτόματα, για τα διάφορα γεγονότα, με βάση τις αλγοριθμικές διαδικασίες που έχουν οριστεί παραπάνω.

Εξετάζουμε τι γίνεται όταν ο ασθενής ζητά το φάρμακο, δηλαδή εισέρχεται στο σύστημα το γεγονός **a**.

OB(a→b)
b→t
-b→-t

-t→d
t→-d
OB(-d)
a

Δεν πληρούνται οι συνθήκες για τον κανόνα Modus Ponens, οπότε σε αυτό το στάδιο δεν προκύπτει άμεσα κάποια νέα πληροφορία. Το σύστημα εξετάζει τι συμβαίνει σε κάθε περίπτωση, ανάλογα με τις διάφορες πράξεις, **b/-b** που μπορεί να κάνει.

Περίπτωση Πρώτη

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b
t

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης της πράξης **b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
b→t
-b→-t

-t→d
t→-d
OB(-d)
a
b
t
-d

Όπου δεν υπάρχει καμία αντίφαση.

Περίπτωση δεύτερη

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b

και σε επόμενο στάδιο πάλι, από Modus Ponens το **d** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b
-t

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης της πράξης **-b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)

b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b
-t
d

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε δύο σημεία,

OB(a→b) (πρώτο σημείο)
b→t
-b→-t
-t→d
t→-d
OB(-d) (δεύτερο σημείο)
a (πρώτο σημείο)
-b (πρώτο σημείο)
-t
d (δεύτερο σημείο)

το ένα αφορά στην υποχρέωση του ρομπότ να δίνει φάρμακο στον ασθενή αν του ζητηθεί από τον ασθενή και το άλλο στην υποχρέωση του ρομπότ να αποφεύγει το να οδηγεί τον ασθενή σε θάνατο.

Σημειώνεται σε αυτό το σημείο ότι με την τέλεση απλώς της πράξης **-b** ήδη υπήρχε λειτουργική αντίφαση που το σύστημα μπορεί να εντοπίσει και έτσι να απορρίψει αυτή την εναλλακτική. Το κατά πόσο η απόρριψη θα γίνεται άμεσα ή μετά την εξαγωγή όλων των συνεπειών, είναι περισσότερο θέμα σύμβασης. Εν τέλει, επιλέχτηκε να γίνεται μετά την εξαγωγή όλων των συνεπειών και γιατί στα πλαίσια αυτών των μοντέλων η λειτουργία Modus Ponens θεωρείται ότι γίνεται ουσιαστικά άμεσα (ο όρος που χρησιμοποιήθηκε παραπάνω είναι "άμεση φυσική συνέπεια") αλλά και για να μπορούν να αξιολογηθούν οι πιθανές επιπτώσεις του πράκτορα τεχνητής νοημοσύνης σε βάθος.

Εξετάζουμε τώρα τι γίνεται όταν ο ασθενής δεν ζητά το φάρμακο, δηλαδή εισέρχεται στο σύστημα το γεγονός **-a**.

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a

Δεν πληρούνται οι συνθήκες για τον κανόνα Modus Ponens, οπότε σε αυτό το στάδιο δεν προκύπτει άμεσα κάποια νέα πληροφορία. Το σύστημα εξετάζει τι συμβαίνει σε κάθε περίπτωση, ανάλογα με τις διάφορες πράξεις, **b/-b** που μπορεί να κάνει.

Περίπτωση πρώτη

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b
t

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης της πράξης **b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b
t
-d

Όπου δεν υπάρχει καμία αντίφαση. Όμως, παρόλο που δεν υπάρχει αντίφαση, υπάρχει ένα σημείο που ξεκάθαρα θα ήταν μη επιθυμητό για το μέσω άνθρωπο και συμβαίνει λόγω του ότι το σύστημα, όπως έχει οριστεί μέχρι στιγμής απλά εξασφαλίζει ότι ο άνθρωπος δεν θα πεθάνει, αγνοώντας τους υπόλοιπους παράγοντες που θα μπορούσαν να θεωρηθούν σχετικοί. Συγκεκριμένα, το σύστημα αποφεύγει ασυνέπειες, δίνοντας στον ασθενή ένα φάρμακο που δεν ήθελε, κάτι που στις περισσότερες περιπτώσεις θα θεωρούταν καταπάτηση των δικαιωμάτων του. Εν τέλει το μέχρι πιο σημείο θέλει ο εκάστοτε μελετητής ένα θεωρητικό σύστημα να καλύπτει τις διάφορες περιπτώσεις είναι θέμα που αφορά τον μελετητή.

Στα πλαίσια αυτής της εργασίας θα υπάρξει περαιτέρω ανάπτυξη, σε κάποιο βαθμό, σαφώς όμως θα υπάρχουν παράγοντες που εν τέλει θα αγνοηθούν.

Περίπτωση δεύτερη

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
-t

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης της πράξης **-b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
-t
d

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε ένα σημείο,

OB(a→b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
-t
d

το οποίο αφορά στην υποχρέωση του ρομπότ να φροντίζει ο ασθενής να μεν οδηγείται σε θάνατο.

Το ρομπότ θα ελέγξει τους δύο "κόσμους" που δημιουργούνται από τις πιθανές επιλογές του και θα επιλέξει τον πρώτο, όπου δεν παρατηρείται αντίφαση.

3.1.3 Τρίτο Ιατρικό Μοντέλο

Όπως όμως έγινε ξεκάθαρο παραπάνω, παρόλο που το σύστημα έχει τη δυνατότητα να κάνει την επιλογή για την οποία είχε σχεδιαστεί, εν τούτοις έχει ακόμα πολύ προφανείς ελλείψεις οι οποίες πρέπει να λυθούν. Στο παρόν στάδιο θα προστεθεί μία πρόταση που θα φροντίσει ακριβώς αυτό που είχε προαναφερθεί, δηλαδή το να μην δύναται να δοθεί με το ζόρι φάρμακο σε έναν ασθενή που δεν το θέλει. Εισάγεται δηλαδή η έννοια της αυτονομίας ασθενούς.

OB(-a→-b)

που σε συνδυασμό με την ήδη υπάρχουσα πρόταση

OB(a→b)

βεβαιώνει εν γένει, ότι αυτό που θέλει ο ασθενής, όσον αφορά το κατά πόσον θα πάρει φάρμακο ή όχι, είναι υποχρεωτικό.

Το σύστημα έχει πλέον την παρακάτω μορφή

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)

όπως έχει συμβεί μέχρι τώρα, θα παρουσιαστούν αναλυτικά οι διάφορες μεταβάσεις που συμβαίνουν για τα διάφορα εξωτερικά γεγονότα και τις πράξεις του συστήματος, που παραμένουν όπως μέχρι τώρα

Πράξεις: **b/-b**, δηλαδή το ρομπότ, μπορεί να δώσει ή να μην δώσει ένα φάρμακο.
Γεγονότα: **a/-a**, δηλαδή μπορεί ο ασθενής να ζητήσει ή να μην ζητήσει το φάρμακο.

Εξετάζεται όπως και παραπάνω τι γίνεται για τα διάφορα γεγονότα και πράξεις, ξεκινώντας από το γεγονός **a**, δηλαδή ότι ο ασθενής ζητά το φάρμακο. Το σύστημα μεταβαίνει σε κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a

Δεν προκύπτουν νέες προτάσεις λόγω Modus Ponens, οπότε το ρομπότ θα εξετάσει τι κόσμοι προκύπτουν για τις δύο πιθανές πράξεις που μπορεί να κάνει.

Περίπτωση Πρώτη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

OB(a→b)
OB(-a→-b)

b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b
t

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης της πράξης **b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
b
t
-d

όπου δεν υπάρχει καμία αντίφαση, είναι ένας αποδεκτός κόσμος.

Περίπτωση Δεύτερη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)

a
-b

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b
-t

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης της πράξης **-b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
a
-b
-t
d

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε δύο σημεία,

OB(a→b) (σημείο πρώτο)
OB(-a→-b)

b→t
-b→-t
-t→d
t→-d
OB(-d) (σημείο δεύτερο)
a (σημείο πρώτο)
-b (σημείο πρώτο)
-t
d (σημείο δεύτερο)

το ένα αφορά στην υποχρέωση του ρομπότ να δίνει φάρμακο στον ασθενή αν του ζητηθεί από τον ασθενή και το άλλο στην υποχρέωση του ρομπότ να αποφεύγει το να οδηγεί τον ασθενή σε θάνατο.

Το ρομπότ θα ελέγξει τους δύο "κόσμους" που δημιουργούνται από τις πιθανές επιλογές του και θα επιλέξει τον πρώτο, όπου δεν παρατηρείται αντίφαση.

Το επόμενο μέρος της ανάλυσης παρουσιάζει περισσότερο ενδιαφέρον. Συγκεκριμένα εξετάζεται τι γίνεται στο μέχρι τώρα σύστημα που έχουμε δομήσει, εάν δεν ζητήσει ο ασθενής το φάρμακο του.

Με την έλευση το εξωτερικού γεγονότος **-a**, το σύστημα περνά στη μορφή

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a

Δεν προκύπτουν νέες προτάσεις λόγω Modus Ponens, οπότε το ρομπότ θα εξετάσει τι κόσμοι προκύπτουν για τις δύο πιθανές πράξεις που μπορεί να κάνει.

Περίπτωση Πρώτη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

OB(a→b)
OB(-a→-b)

b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b
t

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης της πράξης **b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b
t
-d

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε ένα σημείο,

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
b
t
-d

που αφορά στην αυτονομία του ασθενούς. Άρα, αυτός ο κόσμος θα απορριφθεί.

Περίπτωση Δεύτερη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
-t

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης της πράξης **-b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)

-a
-b
-t
d

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε ένα σημείο,

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
-t
d

το οποίο αφορά στην υποχρέωση του ρομπότ να φροντίζει ο ασθενής να μεν οδηγείται σε θάνατο. Άρα, αυτός ο κόσμος θα απορριφθεί.

Για πρώτη φορά στην ανάπτυξη των μοντέλων που έχει γίνει μέχρι τώρα εμφανίζεται μια ιδιαίτερη κατάσταση. Από τη στιγμή της έλευσης του εξωτερικού γεγονότος **-a** ήταν εξασφαλισμένο, ότι το σύστημα, οτιδήποτε και να έκανε, θα κατέληγε σε λειτουργική αντίφαση.
Αν κάνουμε ένα βήμα πίσω και δούμε τι συμβαίνει σε ένα γενικότερο επίπεδο, αυτό που παρατηρείται είναι η αδυναμία ενός πράκτορα εφοδιασμένου με ένα σύστημα τεχνητής νοημοσύνης να ανταποκριθεί στην αποστολή που του ανατέθηκε. Όπως είχε αναφερθεί παραπάνω τέτοιοι πράκτορες είναι προτιμότερο να έχουν βοηθητικό χαρακτήρα παρά πλήρη κυριότητα σε μία κατάσταση. Άρα, πολύ εύλογο συμπέρασμα είναι ότι σε μια τέτοια περίπτωση το ρομπότ θα "ζητήσει βοήθεια" ενημερώνοντας τον υπεύθυνο γι' αυτό άνθρωπο και καλώντας αυτόν να επιληφθεί της κατάστασης.
Φυσικά, από σχεδιαστικής απόψεως θα μπορούσε κάποιος να θεωρήσει ότι ένα πιο αναπτυγμένο σύστημα δεν θα αντιμετώπιζε αυτό το πρόβλημα. Ενώ αυτό μπορεί να ισχύει, και όντως, σε επόμενα στάδια της εργασίας θα παρουσιαστούν πιο αναπτυγμένα συστήματα που δεν "εγκλωβίζονται" όπως το παραπάνω, θα πρέπει να υπογραμμιστεί ότι όσο περισσότερες μεταβλητές προσθέτονται στο σύστημα, τόσο πιο πιθανό γίνεται να προκύπτουν καταστάσεις που θα εμπεριέχουν αναγκαστικά αντιφάσεις τις οποίες ένας πράκτορας τεχνητής νοημοσύνης δεν θα μπορεί να λύσει.
Αυτό δεν είναι κάτι που θα πρέπει να αποθαρρύνει την ανάπτυξη τέτοιων συστημάτων. Το να μπορούν να λύνουν ένα μεγάλο μέρος των περιπτώσεων και να χρειάζονται εξωτερική παρέμβαση για κάποιες είναι μία αποδεκτή κατάσταση.
Όσον αφορά τα πλαίσια αυτής της διπλωματικής εργασίας, των μοντέλων και αλγορίθμων που παρουσιάζει επιλέχτηκε η κατάσταση κατά την οποία δεν μπορεί να προχωρήσει με τίποτα το σύστημα να επιλύεται εξωαλγοριθμικά, χωρίς την κωδικοποιημένη παρουσία μιας υποσυνθήκης που για παράδειγμα θα μπορούσε να εμπεριέχει σε deontic logic προτάσεις που θα οδηγούσαν σε κλήση του επιβλέποντα το συστήματος.

Αυτό συμβαίνει για να υπογραμμιστεί ότι εφόσον μιλάμε για μοντέλα συμπεριφοράς τεχνητής νοημοσύνης, πάντα θα υπάρχουν όρια τα οποία δεν θα μπορούν να ξεπεραστούν.

3.1.3.1 Τρίτο Ιατρικό Μοντέλο-Εναλλακτικές

Παρόλο που όπως αναφέρθηκε παραπάνω επιλέγεται να αντιμετωπίζεται το "κενό" του συστήματος εξωαλγοριθμικά για τους λόγους που αναφέρθηκαν, θα αναφερθούν συνοπτικά κάποιες ιδέες για μικρο βελτιώσεις που θα μπορούσαν να προσφέρουν μια διέξοδο στο σύστημα χωρίς να οδηγούν σε τερματισμό του αλγόριθμου.

- "Σπάσιμο" των γεγονότων **a/-a** και των πράξεων **b/-b** σε αλυσίδες γεγονότων **a[1-κ]/-a[1-κ]** και αντίστοιχα αλυσίδες πράξεων **b[1-κ]/-b[1-κ]**, όπου κ το πλήθος των επαναλαμβανόμενων ερωταπαντήσεων μεταξύ ασθενούς και ρομπότ. Η ιδέα είναι πως με τις επαναλήψεις ο ασθενής έχει τη δυνατότητα να αλλάξει τη γνώμη του.
- Εσωτερίκευση της κλήσης επιβλέποντα σαν πράξη του συστήματος.
- Παρουσίαση πληροφοριών στον ασθενή για τη σημασία του φαρμάκου και δεύτερη ερώτηση.
- Έλεγχος της παρούσας κατάστασης του ασθενούς, π.χ. λήψη αίματος και μετέπειτα επαναπροσδιορισμός του αν όντος χρειάζεται φάρμακο.

Θα παρουσιαστεί η υλοποίηση μίας από αυτές τις περιπτώσεις, αυτή της εσωτερίκευσης της κλήσης επιβλέποντα παρόλο που όπως εξηγήθηκε παραπάνω είναι προτιμότερο τέτοιου είδους γεγονότα να αντιμετωπίζονται εκτός του βασικού αλγορίθμου για να τονίζεται πως συστήματα τεχνητής νοημοσύνης θα χρειάζονται βοήθεια από ανθρώπους σε κάποια σημεία που ενδέχεται να ξεφεύγουν των δυνατοτήτων τους.

Το σύστημα έχει πλέον την παρακάτω μορφή

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)

όπου "||" είναι το λογικό "ή"

Πράξεις: **b/-b**, δηλαδή το ρομπότ, μπορεί να δώσει ή να μην δώσει ένα φάρμακο. **n/-n**, δηλαδή το ρομπότ μπορεί να καλέσει ή να μην καλέσει τον επιβλέποντα ιατρό
Γεγονότα: **a/-a**, δηλαδή μπορεί ο ασθενής να ζητήσει ή να μην ζητήσει το φάρμακο.

Εξετάζεται όπως και παραπάνω τι γίνεται για τα διάφορα γεγονότα και πράξεις, ξεκινώντας από το γεγονός **a**, δηλαδή ότι ο ασθενής ζητά το φάρμακο. Το σύστημα μεταβαίνει σε κατάσταση

OB(a→b)
OB(-a→-b)

b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a

Δεν προκύπτουν νέες προτάσεις λόγω Modus Ponens, οπότε το ρομπότ θα εξετάσει τι κόσμοι προκύπτουν για τους τέσσερις πιθανούς συνδυασμούς πράξεων που μπορεί να κάνει.

Περίπτωση Πρώτη-Δεύτερη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
b
n/-n

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
b
n/-n

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
b
t

n/-n

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης της πράξης **b** καθώς και **n/-n** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
b
t
n/-n
-d

όπου δεν υπάρχει καμία αντίφαση, είναι ένας αποδεκτός κόσμος. Προσθέσαμε την πρόταση **n/-n** , αντί να εξετάσουμε ξεχωριστά για τις δύο πράξεις, γιατί στην προκειμένη περίπτωση θα συνέβαινε ακριβώς το ίδιο είτε είχαμε **n** είτε **-n**.

Περίπτωση Τρίτη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
n

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
n

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
-t
n

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης των πράξεων **-b,n** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
-t
d
n

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση στο σημείο,

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
-t
d
n

που αφορά στην υποχρέωση του ρομπότ να δίνει φάρμακο στον ασθενή αν του ζητηθεί από τον ασθενή.

Περίπτωση Τέταρτη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
-n

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
-n

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a
-b
-t
-n

άρα με την έλευση της εξωτερικής πληροφορίας **a** και της μετέπειτα τέλεσης των πράξεων **-b,n** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
a

-b
-t
d
-n

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε δύο σημεία,

OB(a→b) (σημείο πρώτο)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n) (σημείο δεύτερο)
a (σημείο πρώτο)
-b (σημείο πρώτο)
-t
d (σημείο δεύτερο)
-n (σημείο δεύτερο)

το ένα αφορά στην υποχρέωση του ρομπότ να δίνει φάρμακο στον ασθενή αν του ζητηθεί από τον ασθενή και το άλλο στην υποχρέωση του ρομπότ να αποφεύγει το να οδηγεί τον ασθενή σε θάνατο.

Το ρομπότ θα ελέγξει τους τέσσερις "κόσμους" που δημιουργούνται από τις πιθανές επιλογές του και θα επιλέξει ή τον πρώτο ή τον δεύτερο, όπου δεν παρατηρείται αντίφαση. Ως προς ποιον από τους δύο θα επιλέξει είναι θέμα σύμβασης. Μπορεί να προτιμάται αυτός με **-n** γιατί τελούνται ουσιαστικά λιγότερες πράξεις ή αυτός με **n** γιατί θεωρείται χρήσιμο να ενημερώνεται ούτως ή άλλως ο ιατρός. Εξετάζεται παρακάτω η άλλη ομάδα περιπτώσεων

Με την έλευση το εξωτερικού γεγονότος **-a**, το σύστημα περνά στη μορφή

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a

Δεν προκύπτουν νέες προτάσεις λόγω Modus Ponens, οπότε το ρομπότ θα εξετάσει τι κόσμοι προκύπτουν για τις τέσσερις πιθανές πράξεις που μπορεί να κάνει.

Περίπτωση Πρώτη-Δεύτερη

OB(a→b)
OB(-a→-b)
b→t
-b→-t

-t→d
t→-d
OB(-d||n)
-a
b
n/-n

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
b
n/-n

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
b
t
n/-n

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης της πράξης **b** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
b
t
-d
n/-n

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε ένα σημείο

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
b
t
-d
n/-n

που αφορά στην αυτονομία του ασθενούς. Άρα, αυτός ο κόσμος θα απορριφθεί.

Περίπτωση Τρίτη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
n

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
-b
n

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t

-t→d
t→-d
OB(-d||n)
-a
-b
-t
n

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης των πράξεων **-b**,**n** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
-b
-t
d
n

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα δεν εντοπίζει αντίφαση

Περίπτωση Τέταρτη

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d)
-a
-b
-n

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)

-a
-b
-n

και σε επόμενο στάδιο, πάλι από Modus Ponens το **d** , αφού

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
-b
-t
-n

άρα με την έλευση της εξωτερικής πληροφορίας **-a** και της μετέπειτα τέλεσης των πράξεων **-b** και **-n** το σύστημα καταλήγει στην κατάσταση

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
-b
-t
d
-n

Ελέγχοντας το σύνολο των προτάσεων, το σύστημα εντοπίζει λειτουργική αντίφαση σε ένα σημείο,

OB(a→b)
OB(-a→-b)
b→t
-b→-t
-t→d
t→-d
OB(-d||n)
-a
-b
-t
d
-n

που αφορά στην υποχρέωση του ρομπότ να αποφεύγει ο ασθενής να οδηγείται σε θάνατο.

Το ρομπότ θα ελέγξει τους τέσσερις "κόσμους" που δημιουργούνται από τις πιθανές επιλογές του και θα επιλέξει τον τρίτο, όπου δεν παρατηρείται αντίφαση.

3.1.4 Μέγιστη Ικανοποίηση

Μέχρι στιγμής τα παραδείγματα που μοντελοποιήθηκαν ήταν αρκετά απλοϊκά και η ιδέα της "μέγιστης ικανοποίησης υπό προϋποθέσεις" που παρουσιάστηκε ως βασικός πυλώνας υλοποιήθηκε "αόρατα" μέσω της ίδιας της φύσης του μοντέλου και του παραδείγματος που περιέγραφε. Στο επόμενο στάδιο θα γίνει προσπάθεια πιο εμφανούς παρουσίασης.

Το παράδειγμα που θα δείξουμε είναι γενικό και αφορά στην θεωρητική κατασκευή ενός μοντέλου μέγιστης ευτυχίας.

Πρώτα εξηγείται πως η έννοια της μέγιστης ευτυχίας χτίζεται θεωρητικά.
Το σύστημα καταγράφει τους διάφορους ενδιαφερόμενους, π.χ., για ένα ιατρικό μοντέλο μπορεί να μιλάμε για ασθενή, γιατρό, συγγενής του ασθενούς, κλπ.

Ορίζεται μια γενική μεταβλητή μέγιστης ευτυχίας/ικανοποίησης (max happiness)

Προστίθεται στο σύστημα η πρόταση **OB(max happiness)**

Θεωρείται ότι το σύστημα γνωρίζει με ποιες πράξεις προκαλείται μέγιστη ικανοποίηση
, έστω ότι είναι το σύνολο των πράξεων (x1,x2,x3), δηλαδή **Actions(x1,x2,x3)→max happiness**

Μπορεί αυτό να γίνεται σε ένα ξεχωριστό σύστημα που ο στόχος του είναι απλώς να επιλεχθούν οι πράξεις μέγιστης ευτυχίας.

Η εσωτερική δόμηση είναι τέτοια που να υποστηρίζει ένα άμεσο Modus Ponens

x1&x2&x3→max happiness

όπου "&" είναι το λογικό "και". Ο τρόπος που θα λειτουργεί ένα τέτοιο σύστημα είναι έχοντας υπ' όψιν του κάθε φορά ποιες είναι οι επιθυμητές πράξεις "εξετάζει" διάφορα σύνολα πράξεων μέχρι να πετύχει το σωστό και να "παράξει" μέσω Modus Ponens την μεταβλητή max happiness.

Ο λόγος που η γνώση των επιθυμητών πράξεων δεν θεωρείται ότι παράγει άμεσα το σύνολο που αναζητείται είναι ότι μπορεί να πηγάζει από διαφορετικούς παράγοντες, π.χ. διαφορετικούς ανθρώπους που η ευτυχία του καθενός εξαρτάται από τον αν θα τελεστεί ή όχι μία πράξη. Όπως και παραπάνω στα πλαίσια αυτής της διπλωματικής δεν μας απασχολεί πως αντλούνται μηχανικά οι πληροφορίες.

Ακολουθεί το παραπάνω σε μορφή ψευδοκώδικα

(Only max happiness)

Possible actions: x1/-x1, x2/-x2, x3/-x3
Events: whatever

For all events do
different action set
if max happiness
break
end
end

Αυτό που πρακτικά περιγράφεται είναι ότι με επαναλήψεις καθορίζεται πιο σύνολο πράξεων είναι αυτό που προκαλεί στους ενδιαφερόμενους τη μέγιστη ευτυχία για το κάθε πιθανό γεγονός. Προφανώς εξωγενή γεγονότα μπορεί αλλάζουν τι είναι αυτό που θέλουν κάθε φορά οι ενδιαφερόμενοι.

3.1.5 Μέγιστη Ικανοποίηση υπό Προϋποθέσεις

Στο παρόν στάδιο γίνεται μια προσπάθεια ανάπτυξης μεθοδολογίας για συστηματοποιημένη προσέγγιση της έννοιας της μέγιστης ικανοποίησης υπό προϋποθέσεις, σε αντίθεση με τα παραπάνω παραδείγματα που ως φιλοσοφία χρησιμοποιόταν στον σχεδιασμό του μοντέλου, αλλά δεν υπήρχε τυποποιημένος τρόπος κατασκευής.

Για αυτό το σκοπό θα παρουσιαστούν τρεις διαφορετικές προσεγγίσεις και από αυτές θα καταλήξουμε σε αυτήν που κρίνεται η πιο επιθυμητή.

φφφφφφ

1. Η συνθήκες που δεν πρέπει να παραβιάζονται ορίζονται στη μορφή

OB(a/-a)

προφανώς μπορεί να υπάρχει παραπάνω από μία συνθήκη. Δεν αποτελούν το σύνολο του συστήματος αλλά μόνο ένα μέρος του που περιγράφει τις συνθήκες που δεν πρέπει να καταπατηθούν στην αναζήτηση μέγιστης, υπό προϋποθέσεις ικανοποίησης.

Το σύστημα γνωρίζει από εξωτερική πληροφορία ποιες είναι οι πράξεις που προκαλούν μέγιστη ευτυχία, έστω ένα σύνολο

Max Happiness (x,y,z)

οι συνολικές πράξεις που το σύστημα μπορεί να τελέσει είναι ένα σύνολο πράξεων

Actions x/-x,y/-y,z/-z

επιλέγονται αυτές που προκαλούν μέγιστη ευτυχία και προσθέτονται στο σύστημα ελέγχου συνθηκών.
Παράγονται νέες προτάσεις , αν είναι δυνατόν μέσω Modus Ponens και ελέγχεται αν προκαλούν κάποια αντίφαση με τις συνθήκες του συστήματος ελέγχου. Αν ναι, οι πράξεις επικυρώνονται και περνάνε στο κεντρικό σύστημα. Αν όχι, δοκιμάζεται να αλλαχθεί μία από τις πράξεις και ξαναπροσπαθούμε. Αν αποτύχει η προσπάθεια, αλλάζουμε πάλι μόνο μία από τις πράξεις αφού πρώτα έχουμε επανέρθει στις πράξεις μέγιστης ικανοποίησης. Αν αυτό δεν καταφέρει να αποφύγει αντιφάσεις, όποια πράξη και να αλλάξουμε, αλλάζουμε παραπάνω από μία πράξη τη φορά κ.ο.κ μέχρι να αποφύγουμε αντιφάσεις. Υπάρχει φυσικά πάντα το ενδεχόμενο να μην τα καταφέρουμε

ποτέ.

Ακολουθεί ψευδοκώδικας που περιγράφει τα παραπάνω

```
Actions ← Max happiness Actions
all=number of actions
count =1
while count <=all
        i=1
        while (contradiction & i<=all)
                change count number of actions
                if no contradiction
                        break
                actions ← max happiness actions
                i=i+1
        end
        count=count+1
end
```

Σαν μέθοδος είναι ιδιαίτερα απλοϊκή και εν τέλει δεν επιλέχτηκε. Τα γενικότερα τις σφάλματα είναι δύο.

- Αφενός κινείται τυφλά στις δοκιμές των πράξεων που θα αλλάζουν κάθε φορά, με αποτέλεσμα τον κίνδυνο επαναλήψεων αλλαγών αόριστα. Αυτό μπορεί να διορθωθεί βάζοντας ένα σύστημα μνήμης για τις πράξεις που αλλάζουμε κάθε φορά, ωστόσο αυτό είναι περισσότερη δουλειά απ' ότι τελικά θα χρειαστεί.
- Ένα δεύτερο πρόβλημα που αυτή η μέθοδος δεν καλύπτει είναι ότι αντιμετωπίζει "ισότιμα" τις διάφορες πράξεις παρόλο που η κάθε από αυτές μπορεί να προκαλεί διαφορετική ποσότητα ευτυχίας. Αν π.χ., η επιλογή δύο πράξεων **a1** και **a2** μαζί από ένα σύνολο **a1,a2,a3** προκαλεί την αντίφαση, το σύστημα τελικά θα επιλέξει είτε **-a1,a2,a3** είτε **a1,-a2,a3** . Όμως, το ένα από τα δύο αυτά σύνολα μπορεί να προκαλεί παραπάνω ευτυχία από το άλλο, ενώ το σύστημα τα βλέπει ως ισότιμα.

2. Μία δεύτερη μέθοδος που πάλι απορρίφθηκε είναι η προσπάθεια εσωτερίκευσης των προτάσεων του συστήματος ελέγχου συνθηκών ως πράξεις που επιλέγει το σύστημα, για να μην υπάρξει κάποια αντίφαση.

Αν δηλαδή το σύστημα ελέγχου συνθηκών έχει μέσα πρόταση της μορφής

OBx

και **x/-x** μια πράξη που μπορεί να τελέσει επιλέγει αυτή που ταυτίζεται με τη συνθήκη. Η επιλογή των υπολοίπων πράξεων που δεν ταυτίζονται με κάποια συνθήκη γίνεται σε επόμενο επίπεδο, πιθανώς με σύγκριση ευτυχιών που προκαλούν. Σαν μέθοδος θεωρείται ιδιαίτερα προβληματική καθώς δεν λαμβάνει υπ' όψιν τις εσωτερικές αλληλεπιδράσεις πράξεων, αν για παράδειγμα μέσω Modus Ponens παράγονται νέες προτάσεις από την επιλογή πράξεων.

3. Η μέθοδος που τελικά επιλέχτηκε έχει μια άμεση προσέγγιση που θεωρείται αποδοτική για την πολυπλοκότητα των μοντέλων που εξετάζονται.

Καταγράφεται η "ποσότητα" ευτυχίας που προκαλεί κάθε σύνολο πιθανών πράξεων. Γίνεται μια ταξινόμηση από αυτό με τη μεγαλύτερη ποσότητα προς αυτό με την μικρότερη.
Επιλέγεται το πρώτο. Ελέγχεται αν η προσθήκη αυτών των πράξεων στο σύνολο προκαλεί αντίφαση ή όχι. Αν δεν προκαλεί κρατιέται, αν προκαλεί εξετάζεται το αμέσως επόμενο.

3.1.6 Τέταρτο Ιατρικό Μοντέλο

Χρησιμοποιώντας την παραπάνω μέθοδο αναπτύσσεται παρακάτω ένα μοντέλο συστήματος μέγιστης ικανοποίησης υπό προϋποθέσεις.

Το σενάριο του συστήματος αφορά πάλι σε ένα βοηθητικό νοσοκομειακό ρομπότ που επιβλέπει έναν ασθενή, όντας συνδεδεμένο σε ένα σύστημα deontic logic. Ο ασθενής πάσχει από μία χρόνια πάθηση, όπως ζάχαρο για την οποία συνιστάται να παίρνει καθημερινά φάρμακο, όμως υπάρχει η δυνατότητα να μην το πάρει για περιορισμένο χρονικό διάστημα αν η κατάστασή του είναι καλή. Ταυτόχρονα, υπάρχει ένας υπεύθυνος αποθήκης που ελέγχει το απόθεμα των φαρμάκων και επιθυμεί αν είναι χαμηλό να μην δίνονται άμεσα φάρμακα, μέχρι να αναπληρωθεί το απόθεμα.
Το ρομπότ καλείται να πάρει διάφορες αποφάσεις σχετικά με το τι θα γίνει τελικά με το φάρμακο, χωρίς να καταπατά τα δικαιώματα του ασθενούς που έχουν οριστεί.

Το σύστημα είναι "χωρισμένο" σε δύο τμήματα, ένα που ασχολείται με το κομμάτι μέγιστης ικανοποίησης, ένα που καλύπτει τις υπόλοιπες λειτουργίες. Στα πλαίσια αυτής της διπλωματικής εργασίας δεν μας απασχολεί πως γίνεται η εσωτερική επικοινωνία των τμημάτων, θεωρείται απλώς ότι η πληροφορία περνά αυτόματα και είναι αξιοποιήσιμη.

Τμήμα Μέγιστης Ικανοποίησης.

Οι δύο ενδιαφερόμενοι είναι ο ασθενής και ο αποθηκάριος.
Η ικανοποίηση τους ποσοτικοποιήται με μονάδες.
Οι μονάδες πηγάζουν ανάλογα με τα εκάστοτε θέλω τους για τα δεδομένα εξωτερικά γεγονότα.
Ο ασθενής "δίνει" μια μονάδα όταν συμφωνεί το ρομπότ με το αίτημά του, ο αποθηκάριος "δίνει" μια μονάδα όταν παρουσία ελλείμματος δεν δίνεται φάρμακο και μία μονάδα όταν ελλείψει ελλείμματος δίνεται φάρμακο.

Πράξεις: **b/-b**, δηλαδή το ρομπότ, μπορεί να δώσει ή να μην δώσει ένα φάρμακο.
Γεγονότα: **a/-a**, δηλαδή μπορεί ο ασθενής να ζητήσει ή να μην ζητήσει το φάρμακο και **e/-e** , δηλαδή μπορεί να υπάρχει πλεόνασμα στη αποθήκη ή έλλειψη.

Οι κανόνες το συστήματος μέγιστης ικανοποίησης υπό προϋποθέσεις είναι

OB(a→b)
OB(-a→-b)

που αποτελούν ουσιαστικά ένα μοντέλο της αυτονομίας του ασθενούς εκφρασμένο με όρους deontic logic όπως είδαμε παραπάνω.

Ακολουθεί παρουσίαση των διάφορων περιπτώσεων.

Περίπτωση 1: **e&a**

Ικανοποίηση αν επιλεγεί **b** είναι 2 (από αυτονομία ασθενούς και απουσία ελλείμματος).
Ικανοποίηση αν επιλεγεί **-b** είναι 0.

Ελέγχεται πρώτα το **b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
e
a
b

δεν υπάρχει αντίφαση και επιλέγεται το **b**

Περίπτωση 2: **-e&a**

Ικανοποίηση αν επιλεγεί **b** είναι 1 (από αυτονομία ασθενούς).
Ικανοποίηση αν επιλεγεί **-b** είναι 1 (παρακράτηση φαρμάκου λόγω έλλειψης).

Ελέγχεται πρώτα το **b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
-e
a
b

δεν υπάρχει αντίφαση και επιλέγεται το **b**

Σημειώνεται ότι όπως έχει οριστεί το σύστημα υπολογισμού πράξεων για μέγιστη ικανοποίηση υπό προϋποθέσεις αν μια πράξη περάσει τον έλεγχο αντίφασης, θα επιλεχτεί, καθώς εξετάζουμε σειριακά από την πρώτη σε κατάταξη ικανοποίησης πράξης και αν δεν υπάρχει αντίφαση κρατιέται. Σε περίπτωση ισοβαθμίας δοκιμάζεται από σύμβαση η "θετική" πράξη (**b** αντί για **-b)**. Το τελικό αποτέλεσμα δεν αλλάζει φυσικά, αν εξετάζαμε πρώτα το **-b** το σύστημα θα το απόρριπτε λόγω παραβίασης του κανόνα αυτονομίας και θα πέρναγε στην επόμενη πράξη, το **b** το οποίο και θα επέλεγε.

Περίπτωση 3: **e&-a**

Ικανοποίηση αν επιλεγεί **b** είναι 1 . (από απουσία ελλείμματος)
Ικανοποίηση αν επιλεγεί **-b** είναι 1 (από αυτονομία ασθενούς).

Ελέγχεται πρώτα το **b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
e
-a
b

εντοπίζεται λειτουργική αντίφαση

OB(a→b)
OB(-a→-b)
e
-a
b

και άρα το **b** απορρίπτεται.

Ελέγχεται τώρα το **-b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
e
-a
-b

δεν υπάρχει αντίφαση και επιλέγεται το **-b**

Περίπτωση 4: **-e&-a**

Ικανοποίηση αν επιλεγεί **b** είναι 0 .
Ικανοποίηση αν επιλεγεί **-b** είναι 2 (από αυτονομία ασθενούς και παρακράτηση φάρμακου λόγω έλλειψης).

Ελέγχεται πρώτα το **-b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
-e
-a
-b

δεν υπάρχει αντίφαση και επιλέγεται το **-b**

Το άλλο τμήμα του συστήματος που ασχολείται με τα υπόλοιπα μέρη του σεναρίου, όπως την αντίδραση του ασθενούς στη λήψη φαρμάκου ή όχι είναι

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)

οι όροι που προστέθηκαν είναι οι εξής

check: Το ρομπότ ελέγχει την παρούσα κατάσταση του ασθενούς.
s: Ο είναι σε τέτοια κατάσταση που η μη λήψη φαρμάκου θα τον οδηγήσει σε θάνατο.

Πέρα από τους υπάρχοντες κανόνες

b→t
-b→-t
t→-d
OB(-d)

που η λειτουργία τους εξηγήθηκε σε προηγούμενα στάδια υπάρχουν οι εξής νέοι (ή τροποποιημένοι)

(-t&s)→d, που εξηγεί ότι η μη λήψη φαρμάκου ενόσω ο ασθενής είναι στην προβληματική κατάσταση **s** θα τον οδηγήσει σε θάνατο. Το **&** έχει το ρόλο του λογικού "και". Ο κανόνας αυτός αντικατέστησε τον προηγούμενο κανόνα **-t→d**.

-t→check, που εξηγεί πως όταν ο ασθενής δεν θέλει το φάρμακο του το ρομπότ θα ελέγξει την κατάσταση της υγείας του. Αντικαθιστά τον προηγούμενο κανόνα **-t→d**.

check→s/-s, που εξηγεί πως με τον έλεγχο της κατάστασης του ασθενούς θα έλθει στο σύστημα μια πληροφορία σχετικά με την κατάσταση του. Η πληροφορία αυτή θεωρείται ότι έρχεται άμεσα, καθώς στα πλαίσια της μοντελοποίησης που επιτελεί η διπλωματική αυτή εργασία δεν αναπαριστάται σε βάθος το υλικό κομμάτι άντλησης και μετάδοσης πληροφορίας ,είτε πρόκειται για αιμοληψία, είτε πρόκειται για ανταλλαγή μηνυμάτων μεταξύ δύο συστημάτων είτε κάτι άλλο. Το / χρησιμοποιείται για να δηλωθεί πως θα γίνει είτε το ένα (**s**) είτε το άλλο (**-s**). Τα δύο αυτά στοιχεία, **s** και **-s** θα λέγονται εσωτερικά δεδομένα.

Όπως και στα προηγούμενα συστήματα θα εξεταστούν οι διάφορες περιπτώσεις. Αντί να τρέχουμε το σύστημα για τα εξωτερικά γεγονότα θα προσθέτουμε μόνο την εκάστοτε πράξη που πήραμε από

το σύστημα μέγιστης ικανοποίησης υπό προϋποθέσεις. Συνεπώς οι προηγούμενες περιπτώσεις 1,2 και 3,4 αντίστοιχα "ενώνονται" ανά δύο σε μία, τις νέες περιπτώσεις 1 και 2 αντίστοιχα.

Περίπτωση 1:**b**

Το **b** εισάγεται στο σύστημα το οποίο καταλήγει στη μορφή

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
b

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s /0(data)
t→-d
OB(-d)
b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** , αφού

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
b
t

άρα με την τέλεσης της πράξης **b** το σύστημα καταλήγει στην κατάσταση

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s

t→-d
OB(-d)
b
t
-d

όπου δεν υπάρχει καμία αντίφαση, είναι ένας αποδεκτός κόσμος. Είναι μάλιστα ο μόνος δυνατός κόσμος στο παρόν σύστημα με την εισαγωγή της πράξης **b** και φυσικά γίνεται αποδεκτός.

Περίπτωση 2:**-b**

Το **-b** εισάγεται στο σύστημα το οποίο καταλήγει στη μορφή

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b

και σε επόμενο στάδιο, το **check** , αφού

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t

το σύστημα περνά πλέον στην κατάσταση

b→t
-b→-t

(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t
check

είναι η πρώτη φορά που το σύστημα καλείται να αντιμετωπίσει μια πρόταση της μορφής

x→x1/x2

ο τρόπος που θα λυθεί είναι χτίζοντας δύο σύνολα, ένα στο οποίο το **x** παράγει το **x1** και ένα που παράγει το **x2**. Με αυτό τον τρόπο βεβαιώνεται η συνέπεια της μεθόδου για κάθε πιθανή πορεία που μπορεί να λάβει το μοντέλο προσπαθώντας να προσεγγίσει τις διάφορες εξελίξεις που θα είχε η κατάσταση στον πραγματικό κόσμο. Όπως εξηγήθηκε παραπάνω η άντληση της πληροφορίας γίνεται άμεσα.

Περίπτωση 2α:**-b**, **check→s**

το σύστημα με την έλευση της πληροφορίας **s**, δηλαδή ότι ο ασθενής είναι σε ιδιαίτερα κακή κατάσταση περνά στην κατάσταση

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t
check
s

Από κανόνα Modus Ponens προκύπτει το **d** , αφού

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t
check
s

αναφέρεται ξανά ότι παρόλο που οι διάφορες προτάσεις του συστήματος παρουσιάζονται η μία κάτω από την άλλη για λόγους οπτικά ευανάγνωστης παρουσίασης, ωστόσο κάθε πρόταση στην πραγματικότητα συνδέεται με τις υπόλοιπες με λογικό "και".

άρα το σύστημα είναι πλέον στην κατάσταση

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t
check
s
d

υπάρχει λειτουργική αντίφαση στο εξής σημείο

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t
check
s
d

που αφορά στην υποχρέωση του ρομπότ να αποφεύγει ο ασθενής να οδηγείται σε θάνατο.

Ο κόσμος αυτός , η παρουσία δηλαδή της πράξης **-b** και του εσωτερικού δεδομένου **s**,οδηγεί αναγκαστικά σε αντίφαση. Άρα το ρομπότ θα καλέσει τον επιβλέποντα για να επιληφθεί αυτός της κατάστασης, καθώς το πρόβλημα που τελικά προέκυψε είναι έξω από τις δυνατότητες του συστήματος του. Επισημαίνεται πάλι ότι έχει επιλεχτεί η ειδοποίηση του επιβλέποντα ιατρού να γίνεται εξωαλγοριθμικά, κάθε φορά που το σύστημα είναι αναπόδραστα οδηγούμενο σε αντίφαση, για να τονιστεί ότι συστήματα τεχνητής νοημοσύνης θα χρειάζονται δυνητικά πάντα και την παρουσία του ανθρώπου για την επίλυση καταστάσεων που δεν δύναται το εκάστοτε σύστημα να προβλέπει.

Περίπτωση 2β:**-b**, **check→-s**

το σύστημα με την έλευση της πληροφορίας **-s**, δηλαδή ότι ο ασθενής δεν είναι σε ιδιαίτερα κακή κατάσταση περνά στην κατάσταση

b→t
-b→-t
(-t&s)→d
-t→check
check→s/-s
t→-d
OB(-d)
-b
-t
check
-s

η οποία είναι μια αποδεκτή κατάσταση στην οποία και θα τερματίσει το σύστημα, δεν υπάρχει κάτι άλλο που χρειάζεται να συμβεί.

3.1.7 Πέμπτο Ιατρικό Μοντέλο

Το επόμενο τμήμα της διπλωματικής εργασίας περιλαμβάνει το τελευταίο στάδιο του ιατρικού μοντέλου για συστήματα που υλοποιούν machine ethics με χρήση deontic logic.
Το μοντέλο αυτό περιλαμβάνει αλληλεπίδραση μεταξύ διαφορετικών συστημάτων με διαφορετικούς στόχους και λειτουργίες το καθένα, που ωστόσο ανταλλάσσουν πληροφορίες και στηρίζονται το ένα στο άλλο. Η λογική υλοποίησης της "συνεργασίας" αυτής θα είναι παρόμοια με αυτή που είχε το παραπάνω ρομπότ για την αξιοποίηση των δύο εσωτερικών λειτουργιών του, αυτής για τη μέγιστη ικανοποίηση υπό προϋποθέσεις και αυτή για τις λοιπές λειτουργίες.

Ακολουθεί η παρουσίαση του σεναρίου που θα μοντελοποιηθεί.

Υπάρχουν τρεις agents (πράκτορες) τεχνητής νοημοσύνης.

- Ο πρώτος θα είναι ο βασικός που επιτελεί τις περισσότερες λειτουργίες και έχει το πιο εκτενές σύστημα. Είναι εφοδιασμένος με ένα διπλό σύστημα, ένα που να υλοποιεί μέγιστη ικανοποίηση υπό προϋποθέσεις και ένα για τις γενικότερες λειτουργίες του. Είναι ο επιβλέπων ενός ασθενούς με μία χρόνια κατάσταση και φροντίζει για τα φάρμακά του. Δίνει το φάρμακο αν ο ασθενής το ζητήσει και αν όχι ελέγχει την κατάστασή του για να δει αν είναι δυνατόν να μη λάβει το φάρμακο χωρίς να κινδυνεύει η ζωή του. Ταυτόχρονα έχει υπ' όψιν του τα θέλω ενός αποθηκάριου που δεν επιθυμεί να δίνονται φάρμακα αν υπάρχει χαμηλό απόθεμα, χωρίς αυτό να του επιτρέπει να καταπατά τα δικαιώματα του ασθενούς.
- Ο δεύτερος πράκτορας αλληλεπιδρά ταυτόχρονα σε δύο ασθενείς, αυτόν που επιβλέπει ο πρώτος και πιθανώς έναν ακόμα, στην εντατική ο οποίος ενδέχεται να χρειάζεται μεταμόσχευση οργάνων. Αν ο πρώτος γίνει κλινικά νεκρός (πληροφορία που πηγάζει από τον πρώτο πράκτορα) και ο ασθενής στην εντατική είναι έτοιμος να δεχτεί μεταμόσχευση ο δεύτερος πράκτορας παίρνει τα όργανα του κλινικά νεκρού δότη.
- Ο τρίτος πράκτορας έχει χαρακτήρα συντονιστή μεταξύ των δύο παραπάνω πρακτόρων και φροντίζει για την σωστή ανταλλαγή πληροφοριών μεταξύ τους. Ταυτόχρονα δίνει πληροφορίες στον ιατρό για τη γενική κατάσταση του ασθενούς. Σημειώνεται ότι αυτές οι πληροφορίες είναι γενικές και διαφέρουν από το πάγωμα του συστήματος και την κλήση

του επιβλέποντος ιατρού σε περίπτωση αναπόδραστης αντίφασης.

Ακολουθεί παρουσίαση υπομνήματος του κάθε πράκτορα καθώς και του συστήματος των.

Πρώτος Πράκτορας

Υπόμνημα

a: Ο ασθενής ζητά φάρμακο.
b: Το ρομπότ δίνει στον ασθενή το φάρμακο.
t: Ο ασθενής παίρνει το φάρμακο.
d: Ο ασθενής οδηγείται σε θάνατο.
e: Υπάρχει μεγάλο απόθεμα φαρμάκων. Η άρνηση αναφέρεται σε μειωμένο απόθεμα, αλλά όχι σε πλήρη έλλειψη.
check: Το ρομπότ ελέγχει την παρούσα κατάσταση του ασθενούς.
s: Ο ασθενής είναι σε τέτοια κατάσταση που η μη λήψη φαρμάκου θα τον οδηγήσει σε θάνατο.
s0: Ο ασθενής είναι κλινικά νεκρός. (θεωρείται μη αναστρέψιμη κατάσταση)
inform: Το ρομπότ ενημερώνει τον ασθενή ότι η κατάσταση του είναι επικίνδυνη και πως η λήψη φαρμάκου είναι απαραίτητη.
ac: Ο ασθενής που έχει ενημερωθεί για την επικίνδυνη του κατάσταση αποδέχεται τελικά το φάρμακο.

Πράξεις: **b/-b**
Λειτουργίες**:check/-check**, **inform/-inform**
Γεγονότα: **a/-a**, **e/-e**, **ac/-ac**
Εσωτερικά δεδομένα: **s/-s/s0**, **t**, **d**

Οι κατηγορίες αυτές σημαίνουν τα εξής

- Πράξεις: Οι δυνατές δράσεις του πράκτορα για τις οποίες καλείται να αποφασίσει αν θα εκτελέσει ή όχι.
- Λειτουργίες: Οι δυνατές δράσεις του πράκτορα που προκύπτουν γραμμικά και απόλυτα με βάση τις εκάστοτε καταστάσεις.
- Γεγονότα: Πληροφορίες που έρχονται εξωτερικά, είτε από άλλους πράκτορες είτε από απαντήσεις ανθρώπων και αντίστοιχες πηγές.
- Εσωτερικά Δεδομένα: Πληροφορίες που ο πράκτορας αποκτά εσωτερικά, σαν αποτέλεσμα των δικών του δράσεων.

Σύστημα

Τμήμα Μέγιστης Ικανοποίησης υπό Προϋποθέσεις

Ob(a→b)
Ob(-a→-b)

Γενικό Τμήμα

b→t
-b→-t

-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac

Οι όροι που προστέθηκαν είναι οι εξής: **s0**, **ac**, **inform**.

Οι κανόνες που προστέθηκαν ή τροποποιήθηκαν είναι οι εξής:
check→s/-s/s0, που εξηγεί ότι ο έλεγχος της κατάστασης ασθενούς πέρα από το να φέρει ο την πληροφορία ότι ο ασθενής είναι σε σχετικά καλή ή σχετικά κατάσταση, μπορεί και να φέρει την πληροφορία ότι είναι κλινικά νεκρός.

s→inform, που εξηγεί ότι αν ο ασθενής είναι σε ιδιαίτερα άσχημη κατάσταση θα ενημερωθεί για αυτό το γεγονός.

inform→ac/-ac, που εξηγεί ότι όταν ο ασθενής ενημερώνεται για το ότι βρίσκεται σε ιδιαίτερα άσχημη κατάσταση, έχει τη δυνατότητα τελικά να δεχθεί το φάρμακο του ή να το αρνηθεί ξανά.

-ac→d, που εξηγεί ότι αν ο ασθενής ενόσω βρίσκεται σε ιδιαίτερα άσχημη κατάσταση αρνηθεί και πάλι το φάρμακο του, οδηγείται δυνητικά σε θάνατο.

ac→-d, που εξηγεί ότι αν ο ασθενής ενόσω βρίσκεται σε ιδιαίτερα άσχημη κατάσταση δεχθεί τελικά το φάρμακο του, δεν οδηγείται δυνητικά σε θάνατο.

t→-s, που εξηγεί ότι αν ο ασθενής πήρε το φάρμακο του θεωρείται ότι είναι σε σχετική καλή κατάσταση, αν δεν έχει διαπιστωθεί το αντίθετο προηγουμένως.

Δεύτερος Πράκτορας

Υπόμνημα

e1: Υπάρχει ασθενής που είναι στην εποπτεία του δευτέρου πράκτορα.
a1: Ο ασθενής είναι σε κατάσταση κατάλληλη για να δεχθεί μεταμόσχευση οργάνων.
a2: Ο ασθενής είναι σε κατάσταση ακατάλληλη για να δεχθεί μεταμόσχευση οργάνων.
h: Άντληση οργάνων από διαθέσιμο δότη.
check1: Έλεγχος κατάστασης ασθενούς.
y1: Υπάρχει κατάλληλος δότης.

Πράξεις: **h/-h**, **check1/-check1**
Γεγονότα: **e1/-e1,y1**
Εσωτερικά δεδομένα: **a1,a2**

Σύστημα Ελέγχου

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2

Σύστημα Άντλησης Οργάνων

OB(a1&y1→h)
OB((a2||-e1||-y1)→-h))

Επιλέχτηκε ο δεύτερος πράκτορας να λειτουργεί σε δύο ξεχωριστά τμήματα. Θα μπορούσε να λειτουργήσει και ενοποιημένα αλλά ο χωρισμός τμημάτων και έχει νόημα σε επίπεδο ρεαλισμού, αφού χωρίζει δύο διαφορετικές διαδικασίες και απλοποιεί τον έλεγχο συνέπειας πάρα πολύ, δεδομένου ότι μας επιτρέπει να εστιάσουμε στις πράξεις και τα γεγονότα που αφορούν στο κάθε τμήμα χωρίς να χρειαζόμαστε εξαντλητική ανάλυση του μεγάλου συνόλου περιπτώσεων που θα είχαμε αν αντιμετωπίζαμε τον δεύτερο πράκτορα ενοποιημένα.

Εξήγηση κανόνων

OB (e1→check1), δηλαδή είναι υποχρεωτικό να ελέγξει την κατάσταση του ασθενούς αν αυτός υπάρχει.

OB (-e1→-check1), δηλαδή είναι υποχρεωτικό να μην προσπαθήσει να ελέγξει την κατάσταση του ασθενούς αν αυτός δεν υπάρχει.

check1→a1/a2, δηλαδή ο έλεγχος της κατάστασης του ασθενούς παράγει πληροφορία σχετικά με το αν ο ασθενής είναι έτοιμος ή όχι για μεταμόσχευση.

OB(a1&y1→h), δηλαδή είναι υποχρεωτικό αν ο ασθενής είναι έτοιμος να δεχτεί μεταμόσχευση και υπάρχει δότης, το ρομπότ να αντλήσει τα όργανα του δότη.

OB((a2||-e1||-y1)→-h)), δηλαδή είναι υποχρεωτικό αν είτε ο ασθενής δεν είναι σε κατάσταση κατάλληλη για να δεχτεί μεταμόσχευση είτε δεν υπάρχει ασθενής που να χρειάζεται μεταμόσχευση, ή δεν υπάρχει δότης να μην προσπαθήσει να αντλήσει τα όργανα του δότη.

Τρίτος Πράκτορας

Υπόμνημα

notify: Ενημέρωση ιατρού για ύπαρξη κλινικά νεκρού ασθενούς.
alert: Ειδοποιεί τον δεύτερο πράκτορα ότι υπάρχει διαθέσιμος δότης.

Πράξεις: **alert/-alert**,**notify/-notify**
Γεγονότα: **s**,**-s**,**s0**
Εσωτερικά δεδομένα: **y1/-y1**

Σύστημα

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1

Εξήγηση κανόνων

OB (s0→alert), δηλαδή αν ο ασθενής του πρώτου πράκτορα είναι κλινικά νεκρός είναι υποχρεωτικό να πληροφορηθεί ο δεύτερος πράκτορας για την ύπαρξη δότη.

OB(s→-alert) και **OB(-s→-alert)** δηλαδή αν ο ασθενής του πρώτου πράκτορα δεν είναι κλινικά νεκρός (είναι δηλαδή σε σχετικά καλή ή ιδιαίτερα κακή κατάσταση) είναι υποχρεωτικό να μην πληροφορηθεί ο δεύτερος πράκτορας για ύπαρξη δότη.

OB(s→-notify) και **OB(-s→-notify)** δηλαδή αν ο ασθενής του πρώτου πράκτορα δεν είναι κλινικά νεκρός (είναι δηλαδή σε σχετικά καλή ή ιδιαίτερα κακή κατάσταση) είναι υποχρεωτικό να μην πληροφορηθεί ο ιατρός πως ο ασθενής είναι κλινικά νεκρός.

OB(s0→ notify), δηλαδή αν ο ασθενής του πρώτου πράκτορα είναι κλινικά νεκρός νεκρός ενημερώνεται για την κατάσταση του πρώτου ασθενούς ο επιβλέποντας ιατρός.

alert→y1, δηλαδή η διαδικασία ενημέρωσης του δεύτερου πράκτορα για την ύπαρξη δότη "γεννά" την κατάλληλη πληροφορία.

-alert→-y1, συμμετρικό του προηγουμένου για να μετατραπεί η απουσία πληροφορίας σε αξιοποιήσιμη πρόταση για το δεύτερο πράκτορα.

Ακολουθεί ένα αναλυτικό παράδειγμα που παρουσιάζει τα παραπάνω σε λειτουργία για τα διάφορα ενδεχόμενα.

Ξεκινάμε από τον πρώτο πράκτορα. Το τμήμα της μέγιστης ικανοποίησης υπό προϋποθέσεις δεν έχει αλλάξει από το προηγούμενο τμήμα.

Οι κανόνες το συστήματος μέγιστης ικανοποίησης υπό προϋποθέσεις είναι

OB(a→b)
OB(-a→-b)

που αποτελούν ουσιαστικά ένα μοντέλο της αυτονομίας του ασθενούς εκφρασμένο με όρους deontic logic όπως είδαμε παραπάνω.

Ακολουθεί παρουσίαση των διάφορων περιπτώσεων.

Περίπτωση 1: **e&a**

Ικανοποίηση αν επιλεγεί **b** είναι 2 (από αυτονομία ασθενούς και απουσία ελλείμματος).
Ικανοποίηση αν επιλεγεί **-b** είναι 0.

Ελέγχεται πρώτα το **b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
e
a
b

δεν υπάρχει αντίφαση και επιλέγεται το **b**

Περίπτωση 2: **-e&a**

Ικανοποίηση αν επιλεγεί **b** είναι 1 (από αυτονομία ασθενούς).
Ικανοποίηση αν επιλεγεί **-b** είναι 1 (παρακράτηση φαρμάκου λόγω έλλειψης).

Ελέγχεται πρώτα το **b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
-e
a
b

δεν υπάρχει αντίφαση και επιλέγεται το **b**

Περίπτωση 3: **e&-a**

Ικανοποίηση αν επιλεγεί **b** είναι 1 . (απουσία ελλείμματος)
Ικανοποίηση αν επιλεγεί **-b** είναι 1 (από αυτονομία ασθενούς).

Ελέγχεται πρώτα το **b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
e
-a

b

εντοπίζεται λειτουργική αντίφαση

OB(a→b)
OB(-a→-b)
e
-a
b

και άρα το **b** απορρίπτεται.

Ελέγχεται τώρα το **-b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
e
-a
-b

δεν υπάρχει αντίφαση και επιλέγεται το **-b**

Περίπτωση 4: **-e&-a**

Ικανοποίηση αν επιλεγεί **b** είναι 0 .
Ικανοποίηση αν επιλεγεί **-b** είναι 2 (από αυτονομία ασθενούς και παρακράτηση φάρμακου λόγω έλλειψης).

Ελέγχεται πρώτα το **-b**

το σύστημα γίνεται

OB(a→b)
OB(-a→-b)
-e
-a
-b

δεν υπάρχει αντίφαση και επιλέγεται το **-b**

Εν τέλει, όπως ήταν αναμενόμενο προκύπτουν δύο ενδεχόμενα, να "παραχθεί" η πληροφορία **b** και ή **-b.**

Εξετάζεται πως προχωρά το σύστημα σε κάθε περίπτωση.

Περίπτωση 1:**b**

Το **b** εισάγεται στο γενικό το οποίο καταλήγει στη μορφή

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
b

Από κανόνα Modus Ponens προκύπτει το **t** , αφού

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **-d** και το **-s** , αφού

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
b
t

άρα με την έλευση της πληροφορίας **b** το σύστημα καταλήγει στην κατάσταση

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
b
t
-d
-s

όπου δεν υπάρχει καμία αντίφαση και ο κόσμος είναι αποδεκτός. Σημειώνεται επίσης ότι παράχθηκε και μία πληροφορία που είναι χρήσιμη για άλλο πράκτορα, το **-s**.

Περίπτωση 2:**-b**

Το **-b** εισάγεται στο γενικό το οποίο καταλήγει στη μορφή

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b

Από κανόνα Modus Ponens προκύπτει το **-t** , αφού

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d

t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b

και σε επόμενο στάδιο, πάλι από Modus Ponens το **check** , αφού

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t

άρα με την έλευση της πληροφορίας **-b** το σύστημα καταλήγει στην κατάσταση

b→t
-b→-t
-ac→d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check

προχωράμε σε ανάλυση του συστήματος για τα διάφορα γεγονότα που μπορεί να παράξει η δράση **check**.

Περίπτωση 2α:**-b**, **check→s**

το σύστημα με την έλευση της πληροφορίας **s**, δηλαδή ότι ο ασθενής είναι σε ιδιαίτερα κακή κατάσταση περνά στην κατάσταση

b→t
-b→-t
-ac→d

ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s

Από κανόνα Modus Ponens προκύπτει το **inform** , αφού

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s

και το σύστημα περνά στην κατάσταση

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform

και εξετάζουμε την επερχόμενη πορεία του συστήματος για τις διάφορες πληροφορίες που μπορεί να προκύψουν από την πράξη **inform**.

Περίπτωση 2αα:**-b**, **check→s**, **inform→ac**

το σύστημα περνά στην κατάσταση

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform
ac

Από κανόνα Modus Ponens προκύπτει το **-d** , αφού

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform
ac

οπότε το σύστημα καταλήγει στην κατάσταση

b→t

-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform
ac
-d

όπου δεν υπάρχει καμία αντίφαση και πρόκειται για αποδεκτό κόσμο. Επίσης, έχει παραχθεί η πληροφορία **s** που μπορούν να αξιοποιήσουν άλλοι πράκτορες.

Περίπτωση 2αβ:**-b**, **check→s**, **inform→-ac**

το σύστημα περνά στην κατάσταση

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform
-ac

Από κανόνα Modus Ponens προκύπτει το **d** , αφού

b→t
-b→-t
-ac→d
-t→check

check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform
-ac

οπότε το σύστημα καταλήγει στην κατάσταση

b→t
-b→-t
-ac→d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s
inform
ac
d

όπου υπάρχει λειτουργική αντίφαση στο σημείο

b→t
-b→-t
-ac→d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
s

inform
ac
d

που αφορά στην υποχρέωση του ρομπότ να μην οδηγείται ο ασθενής σε θάνατο. Το σύστημα σταματά και καλεί επιβλέποντα ιατρό αφού η κατάσταση που κατέληξε ξεφεύγει από τις δυνατότητές του. Επίσης παράχθηκε η πληροφορία **s** που μπορούν να αξιοποιήσουν οι άλλοι πράκτορες.

Περίπτωση 2β:**-b**, **check→-s**

το σύστημα με την έλευση της πληροφορίας **-s**, δηλαδή ότι ο ασθενής δεν είναι σε ιδιαίτερα κακή κατάσταση περνά στην κατάσταση

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform
inform→ac/-ac
-b
-t
check
-s

είναι μια αποδεκτή κατάσταση στην οποία ο πράκτορας δε χρειάζεται να κάνει κάτι άλλο. Επίσης παράγεται η πληροφορία **-s** που μπορούν να τη χρησιμοποιήσουν οι άλλοι πράκτορες.

Περίπτωση 2γ:**-b**, **check→s0**

το σύστημα με την έλευση της πληροφορίας **s0**, δηλαδή ότι ο ασθενής είναι κλινικά νεκρός περνά στην κατάσταση

b→t
-b→-t
-ac→d
ac→-d
-t→check
check→s/-s/s0
t→-d
t→-s
Ob(-d)
s→inform

inform→ac/-ac
-b
-t
check
s0

είναι μια αποδεκτή κατάσταση στην οποία ο πράκτορας δε χρειάζεται να κάνει κάτι άλλο. Επίσης παράγεται η πληροφορία **s0** που μπορούν να τη χρησιμοποιήσουν οι άλλοι πράκτορες.

Έχουν εξαντληθεί το σύνολο των καταστάσεων στις οποίες μπορεί να περάσει το σύστημα του πρώτου πράκτορα. Θα περάσουμε στους υπόλοιπους.

Δεδομένου ότι ο δεύτερος πράκτορας εξαρτάται από πληροφορίες του τρίτου, στο επόμενο στάδιο εξετάζουμε πρώτα τον τρίτο πράκτορα.

Θα εξεταστούν οι διάφορες περιπτώσεις για τις πληροφορίες που έστειλε ο πρώτος πράκτορας στον τρίτο ο οποίος θα τις χειριστεί σαν εξωτερικά γεγονότα.
Περίπτωση 1: **s**

Το **s** εισάγεται σύστημα το οποίο καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s

Το σύστημα έχει τέσσερις πιθανούς συνδυασμούς πράξεων που μπορεί να κάνει

α) **alert&notify**
β) **alert&-notify**
γ) **-alert&notify**
δ) **-alert&-notify**

Ελέγχεται τι γίνεται σε για κάθε συνδυασμό.

Περίπτωση 1α: **s, alert&notify**

με την προσθήκη της εξωτερικής πληροφορίας **s** και των πράξεων **alert** και **notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1

-alert→-y1
s
alert
notify
y1

λόγω **alert→y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω δύο σημείων

OB (s0→alert)
OB(s→-alert) (πρώτο σημείο)
OB(-s→-alert)
OB(s→-notify) (δεύτερο σημείο)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s (πρώτο και δεύτερο σημείο)
alert (πρώτο σημείο)
notify (δεύτερο σημείο)
y1

άρα η επιλογή αυτή θα απορριφθεί.

Περίπτωση 1β: **s, alert&-notify**

με την προσθήκη της εξωτερικής πληροφορίας **s** και των πράξεων **alert** και **-notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s
alert
-notify
y1

λόγω **alert→y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (s0→alert)
OB(s→-alert)

OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s
alert
-notify
y1

άρα η επιλογή αυτή θα απορριφθεί.

Περίπτωση 1γ: **s, -alert&notify**

με την προσθήκη της εξωτερικής πληροφορίας **s** και των πράξεων **-alert** και **notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s
-alert
notify
-y1

λόγω **-alert→-y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s
-alert
notify
-y1

Περίπτωση 1δ: **s, -alert&-notify**

με την προσθήκη της εξωτερικής πληροφορίας **s** και των πράξεων **-alert** και -**notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s
-alert
-notify
-y1

λόγω **-alert→-y1** και Modus Ponens

όπου δεν υπάρχει κάποια λειτουργική αντίφαση και είναι επιλογή που τελικά θα επικυρώσει ο τρίτος πράκτορας αν δεχθεί το εξωτερικό γεγονός **s**.

Περίπτωση 2: **-s**

Το **s** εισάγεται σύστημα το οποίο καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
-s

Το σύστημα έχει τέσσερις πιθανούς συνδυασμούς πράξεων που μπορεί να κάνει

α) **alert&notify**
β) **alert&-notify**
γ) **-alert&notify**
δ) **-alert&-notify**

Ελέγχεται τι γίνεται σε για κάθε συνδυασμό.

Περίπτωση 2α: **-s, alert&notify**

με την προσθήκη της εξωτερικής πληροφορίας **-s** και των πράξεων **alert** και **notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)

OB(-s→-alert)
OB(s0→ notify)
alert→y1
-alert→-y1
-s
alert
notify
y1

λόγω **alert→y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω δύο σημείων

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert) (πρώτο σημείο)
OB(s→-notify)
OB(-s→-notify) (δεύτερο σημείο)
OB(s0→ notify)
alert→y1
-alert→-y1
-s (πρώτο και δεύτερο σημείο)
alert (πρώτο σημείο)
notify (δεύτερο σημείο)
y1

άρα η επιλογή αυτή θα απορριφθεί.

Περίπτωση 2β: **-s, alert&-notify**

με την προσθήκη της εξωτερικής πληροφορίας **-s** και των πράξεων **alert** και **-notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
-s
alert
-notify
y1

λόγω **alert→y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
-s
alert
-notify
y1

άρα η επιλογή αυτή θα απορριφθεί.

Περίπτωση 2γ: **-s, -alert&notify**

με την προσθήκη της εξωτερικής πληροφορίας **-s** και των πράξεων **-alert** και **notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
-s
-alert
notify
-y1

λόγω **-alert→-y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
-s
-alert
notify
-y1

Περίπτωση 2δ: **-s, -alert&-notify**

με την προσθήκη της εξωτερικής πληροφορίας **-s** και των πράξεων **-alert** και **-notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
-s
-alert
-notify
-y1

λόγω **-alert→-y1**

όπου δεν υπάρχει κάποια λειτουργική αντίφαση και είναι επιλογή που τελικά θα επικυρώσει ο τρίτος πράκτορας αν δεχθεί το εξωτερικό γεγονός **-s**.

Περίπτωση 3: **s0**

Το **s** εισάγεται σύστημα το οποίο καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s0

Το σύστημα έχει τέσσερις πιθανούς συνδυασμούς πράξεων που μπορεί να κάνει

α) **alert&notify**
β) **alert&-notify**
γ) **-alert&notify**
δ) **-alert&-notify**

Ελέγχεται τι γίνεται σε για κάθε συνδυασμό.

Περίπτωση 3α: **s0, alert&notify**

με την προσθήκη της εξωτερικής πληροφορίας **s0** και των πράξεων **alert** και **notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s0
alert
notify
y1

λόγω **alert→y1** και Modus Ponens
όπου δεν υπάρχει λειτουργική αντίφαση.

Περίπτωση 3β: **s0, alert&-notify**

με την προσθήκη της εξωτερικής πληροφορίας **s0** και των πράξεων **alert** και **-notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s0
alert
-notify
y1

λόγω **alert→y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s0
alert
-notify
y1

άρα η επιλογή αυτή θα απορριφθεί.

Περίπτωση 3γ: **-s, -alert&notify**

με την προσθήκη της εξωτερικής πληροφορίας **s0** και των πράξεων **-alert** και **notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s0
-alert
notify
-y1

λόγω **-alert→-y1** και Modus Ponens

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify)
alert→y1
-alert→-y1
s0
-alert
notify
-y1

άρα η επιλογή αυτή θα απορριφθεί.

Περίπτωση 3δ: **s0, -alert&-notify**

με την προσθήκη της εξωτερικής πληροφορίας **s0** και των πράξεων **-alert** και **-notify** το σύστημα καταλήγει στη μορφή

OB (s0→alert)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)

OB(s0→ notify)
alert→y1
-alert→-y1
s0
-alert
-notify
-y1

λόγω **-alert→-y1** και Modus Ponens

όπου υπάρχει κάποια λειτουργική αντίφαση σε δύο σημεία λόγω

OB (s0→alert) (πρώτο σημείο)
OB(s→-alert)
OB(-s→-alert)
OB(s→-notify)
OB(-s→-notify)
OB(s0→ notify) (δεύτερο σημείο)
alert→y1
-alert→-y1
s0 (πρώτο και δεύτερο σημείο)
-alert (πρώτο σημείο)
-notify (δεύτερο σημείο)
-y1

άρα ο η επιλογή αυτή θα απορριφθεί και το σύστημα κρατά απλώς την πρώτη επιλογή.

Ανάλογα με το ποια επιλογή προκύπτει από τον τρίτο πράκτορα, έχουμε πιθανώς την πληροφορία **y1** ή **-y1** που θα αξιοποίηση ο δεύτερος πράκτορας ή όχι.

Το τελευταίο τμήμα του ιατρικού μοντέλου είναι η ανάλυση του δεύτερου πράκτορα που όπως αναφέρθηκε παραπάνω χωρίζεται σε δύο τμήματα που το καθένα επικεντρώνεται σε κάποια μόνο από τα γεγονότα και τις πράξεις.

Πράξεις: **h/-h**, **check1/-check1**
Γεγονότα: **e1/-e1, y1/-y1**
Εσωτερικά δεδομένα: **a1,a2**

Σύστημα Ελέγχου

Το τμήμα αυτό αφορά στα γεγονότα **e1/-e1** και στις πράξεις **check1/-check1**

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2

Έστω ότι έρχεται ως πληροφορία το γεγονός **e1**

Περίπτωση 1 **e1, check1**

Το σύστημα είναι στην κατάσταση

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
e1
check1

και άμεσα προκύπτει και η πληροφορία **a1** ή **a2** λόγω **check1→a1/a2** και Modus Ponens άρα καταλήγει το σύστημα στην κατάσταση

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
e1
check1
a1/a2

που είναι αποδεκτή κατάσταση.

Περίπτωση 2 **e1, -check1**

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
e1
-check1

όπου υπάρχει λειτουργική αντίφαση λόγω

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
e1
-check1

και η επιλογή αυτή απορρίπτεται.

Έστω ότι έρχεται ως πληροφορία το γεγονός **-e1**

Περίπτωση 1 **-e1, check1**

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
-e1
check1

και άμεσα προκύπτει και η πληροφορία **a1** ή **a2** λόγω **check1→a1/a2** και Modus Ponens

ενώ υπάρχει λειτουργική αντίφαση λόγω

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
-e1
check1
a1/a2

και η επιλογή αυτή απορρίπτεται.

Περίπτωση 2 **-e1, -check1**

OB (e1→check1)
OB (-e1→-check1)
check1→a1/a2
-e1
-check1

είναι αποδεκτή κατάσταση.

Σύστημα Άντλησης Οργάνων

OB((a1&y1&e1)→h)
OB((a2||-e1||-y1)→-h))

Το τμήμα αυτό αφορά στα γεγονότα **y1/-y1**, **e1/-e1** και **a1,a2** (εσωτερικά δεδομένα του προηγουμένου τμήματος) και στις πράξεις **h/-h**

Αυτή τη φορά οι κανόνες του συστήματος είναι πιο εκτενείς στους όρους που μεταχειρίζονται πράγμα λογικό αν αναλογιστεί κανείς τη βαρύτητα του θέματος που αφορά, συγκεκριμένα την άντληση οργάνων από ένα κλινικά νεκρό άτομο.

Αντί να παρουσιαστούν όλοι οι πιθανοί συνδυασμοί γεγονότων και πράξεων προτιμάται να εξηγηθεί πως "κλειδώνεται" η συνέπεια του συστήματος τμηματικά. Διαφορετικά θα γινόταν μια παρουσίαση πληθώρας περιπτώσεων που δεν έχουν ιδιαίτερη σημασία.

Οι λόγοι για να μην έχουμε άντληση οργάνων θα είναι να μην υπάρχει δότης (**-y1**), να μην υπάρχει ασθενής που να χρειάζεται όργανα (**-e1**) ή να μην είναι ο ασθενής αυτός σε κατάσταση κατάλληλη να δεχτεί τα όργανα(**a2**).

Αν οποιαδήποτε από αυτές τις συνθήκες ισχύουν ο το σύστημα περνά σε μορφή

OB((a1&y1&e1)→h)
OB(x1→-h)
x1

όπου από τον ορισμό της λειτουργικής αντίφασης η τέλεση της πράξης **h** θα οδηγούσε σε αντίφαση, αφού

OB((a1&y1&e1)→h)
OB(x1→-h)
x1
h

άρα η ισχύ οποιασδήποτε των παραπάνω συνθηκών εξασφαλίζει την πράξη **-h**, καθώς με **-h** δεν υπάρχει αντίφαση

OB((a1&y1&e1)→h)
OB(x1→-h)
x1
-h

Ο μόνος τρόπος να μην εξασφαλιζόταν εξαρχής ότι η πράξη **-h** θα τελεστεί θα απαιτούσε όλες οι παραπάνω πληροφορίες να μην ισχύουν. Αυτόματα αυτό θα οδηγούσε στα "αντίθετα" γεγονότα της κάθε κατάστασης να ισχύουν όλα, δηλαδή θα υπάρχει δότης (**y1**), ασθενής που να θέλει μεταμόσχευση (**e1**) και θα είναι και σε κατάλληλη κατάσταση για να τη δεχθεί (**a1**).

Αυτό θα σήμαινε ότι το σύστημα περνά στην εξής κατάσταση

OB(x2→h)
OB((a2||-e1||-y1)→-h))
x2

όπου από τον ορισμό της λειτουργικής αντίφασης η τέλεση της πράξης **-h** θα οδηγούσε σε αντίφαση, αφού

OB(x2→h)
OB((a2||-e1||-y1)→-h))
x2
-h

άρα η ισχύ οποιασδήποτε των παραπάνω συνθηκών εξασφαλίζει την πράξη **h**, η οποία δεν προκαλεί αντίφαση

OB(x2→h)
OB((a2||-e1||-y1)→-h))
x2
h

Σε κάθε περίπτωση η συνέπεια του συστήματος είναι εξασφαλισμένη.

3.2 Μοντέλο Αυτοκινήτων

Το επόμενο στάδιο της διπλωματικής εργασίας αφορά και πάλι τη χρήση των προσεγγίσεων που περιγράφηκαν παραπάνω αλλά αυτή τη φορά για μοντελοποίηση κάτι τελείως διαφορετικού, συγκεκριμένα μιας προσπάθειας για εξομοίωση ενός θεωρητικού συστήματος για αυτοκίνητα που παίρνουν τα ίδια αποφάσεις για την κίνησή τους.

Ο τρόπος που προσεγγίστηκε αυτό είναι σπάζοντας τις επιμέρους λειτουργίες του αυτοκινήτου σε διαφορετικούς τομείς. Ο κάθε τομέας θα θεωρείται ως ένας ξεχωριστός πράκτορας (agent) εφοδιασμένος με ένα σύστημα τεχνητής νοημοσύνης που εξομοιώνεται με ένα σύνολο προτάσεων εκφρασμένων σε deontic logic.

Υποθέτουμε ότι υπάρχει ένας επιβλέπων που έχει πρόσβαση στα δεδομένα όλων των αισθητήρων και με βάση αυτά επιλέγει ποιος τομέας πρέπει να είναι σε ισχύ κάθε φορά. Θα παρουσιαστεί ο τρόπος επιλογής του επιμέρους τομέα και μετά η λειτουργία του κάθε μέρους ξεχωριστά.

Οι τομείς είναι

- **Start processes (Διαδικασίες Εκκίνησης)** που αφορά στις διαδικασίες που πρέπει να κάνει το αμάξι για να ξεκινήσει ή να σταματήσει.
- **Collision processes (Διαδικασίες Σύγκρουσης)** που αφορά στις διαδικασίες που πρέπει να κάνει το αμάξι για να αποφύγει συντριβή.
- **Fuel processes (Διαδικασίες Καυσίμου)** που αφορά στις διαδικασίες που πρέπει να κάνει το αμάξι για να διατηρήσει το καύσιμα του.
- **Destination processes (Διαδικασίες Προορισμού)** που αφορά στις διαδικασίες που πρέπει να κάνει το αμάξι για την ολοκλήρωση μιας διαδρομής.

Destination processes

Αυτή είναι η κατάσταση στην οποία θα βρίσκεται την περισσότερη ώρα το αμάξι και ρυθμίζει την επιλογή διαδρομής, καθώς και τον έλεγχο αν έχουμε φτάσει στον προορισμό ή όχι. Με τον ίδιο τρόπο που στα ιατρικά μοντέλα, δεν απασχολούσε την έρευνα της διπλωματικής με ποιον τρόπο αντλεί μηχανικά το ρομπότ πληροφορίες για την υγεία του ασθενούς, μέσω π.χ. μιας αιμοληψίας, έτσι και εδώ δεν απασχολεί με ποιον τρόπο γίνεται η επιβεβαίωση ότι έχουμε φτάσει στον προορισμό. Σε κάθε στιγμή οι αισθητήρες ελέγχουν και παράγουν την κατάλληλη πληροφορία που περνά στο σύστημα μας.

Οι διαδικασίες που καλείται να επιτελεί συγκεκριμένα ο πράκτορας του τομέα είναι να λάβει καινούργια διαδρομή αν ο χρήστης το επιθυμεί, να συνεχίσει την κίνηση που έχει ήδη αν δεν λάβει καινούργια διαδρομή, ή να παρκάρει αν έχει φτάσει στο στόχο του.
Το μοντέλο που παρουσιάζεται στα πλαίσια αυτής της εργασίας βεβαιώνει ότι την κάθε μια δεδομένη στιγμή το σύστημα θα κάνει την ηθικά σωστή συμπεριφορά. Δεν απασχολεί τα πλαίσια της εργασίας πως μια επόμενη στιγμή η βάση δεδομένων του συστήματος “αδειάζει” από προτάσεις που δεν ισχύουν πια, με τον ίδιο τρόπο που σε προηγούμενα μοντέλα δε μας ενδιέφερε με ποιον τρόπο το ρομπότ που παρακολουθούσε έναν ασθενή έσβηνε από την μνήμη του τις

πληροφορίες που αφορούν στην κατάστασή του μια προηγούμενη μέρα.

Γεγονότα: **new_destination/destination/no_destination**
Το **new_destination** σημαίνει ότι ήρθε καινούργια διαδρομή από τον χρήστη.
Το **destination** σημαίνει ότι δεν έχει έρθει νέα διαδρομή, οπότε το σύστημα κρατά την υπάρχουσα διαδρομή στο στόχο της οποίας δεν έχει φτάσει ακόμα.
Το **no_destination** σημαίνει ότι δεν έχει έρθει νέα διαδρομή και έχουμε φτάσει στο στόχο της υπάρχουσας.
Και οι τρεις αυτοί όροι θεωρούνται εξωτερικά γεγονότα που εισάγονται στην αρχή στο σύστημα, παρόλο τα δύο παράγονται από το σύστημα, όπως θα φανεί παρακάτω. Όπως εξηγήσαμε, το με ποιον τρόπο θα κάνει "εκκαθάριση" μεταξύ κύκλων ξεφεύγει από το ζήτημα που εξετάζει το μοντέλο.

Παρατηρείται επίσης μια ακόμη διαφορά. Μέχρι στιγμής τα διάφορα γεγονότα εξετάζονταν στη μορφή **a/-a**. Αυτό εδώ αλλάζει, καθώς με βάση την υπόθεση του σεναρίου που αναλύεται η ίδια η άρνηση του ενός γεγονότος εμπεριέχεται στην αποδοχή των άλλων. Δεν γίνεται, π.χ. ενώ δεν έχουμε ολοκληρώσει τη διαδρομή να είμαστε κιόλας στο στόχο.

Αντίστοιχη λογική θα ακολουθηθεί και στις πράξεις του πράκτορα όπου εκεί υπάρχει περισσότερο ενδιαφέρον λόγω των νέων συνθηκών που θα προκύψουν από αυτό.

Δεδομένα: **next_destination/next_no_destination** Αυτές είναι η πληροφορίες από τους αισθητήρες του συστήματος για το που το οδήγησαν οι κινήσεις που έγιναν σε αυτό τον κύκλο λειτουργιών. Στον επόμενο, θα αποτελέσουν τα πιθανά εσωτερικά γεγονότα. Όπως και πριν, το πως γίνεται εκκαθάριση συστήματος από ένα κύκλο στον επόμενο δεν απασχολεί όσα μελετώνται στα πλαίσια της εργασίας. Αντίστοιχα δεν απασχολεί το πως μηχανικά γίνεται η μετατροπή πληροφορίας από **next_destination** σε **destination** και από **next_no_destination** σε **no_destination** .

Πράξεις: **move/park**

Το **move** αφορά στην κίνηση που θα κάνει το αμάξι δεδομένου ότι δεν είναι στο στόχο της διαδρομής του. Η κίνηση αυτή φυσικά θα επιφέρει πληροφορίες για το αν έφτασε στο στόχο του ή όχι.
Το **park** αφορά στην διαδικασία παρκαρίσματος που το αμάξι θα κάνει αν έχει φτάσει στο στόχο της διαδρομής.

Εν πρώτοις το σύστημα που προκύπτει από αυτά φαίνεται να χρειάζεται τους εξής κανόνες

OB(no destination→park), που εξηγεί πως είναι υποχρεωτικό αν έχει φτάσει στο στόχο του να παρκάρει.

OB((destination||new_destination))→ move), που εξηγεί πως είναι υποχρεωτικό αν έχει ακόμα διαδρομή μέχρι να καλύψει μέχρι το στόχο του είτε γιατί δεν έχει φτάσει στο στόχο της υπάρχουσας, είτε γιατί ήρθε καινούργια να συνεχίσει να κινείται.

move→next_no_destination/next_destination, που εξηγεί πως η διαδικασία κίνησης "γεννά" πληροφορίες για το κατά πόσο έχουμε φτάσει στο στόχο μας ή όχι. Οι πληροφορίες αυτές θα χρησιμοποιηθούν ως εξωτερικά γεγονότα στο επόμενο στιγμιότυπο λειτουργίας.

Όμως οι κανόνες αυτοί δε φτάνουν. Ο λόγος είναι ο τρόπος που έχει οριστεί η έννοια της λειτουργικής αντίφασης.

OB(a→b)
a
-b

ιδανικά το σύστημα θα πρέπει να απορρίπτει κόσμους που καταπατούν τις υποχρεώσεις του. Όμως αυτό δεν μπορεί να γίνει αν δεν δημιουργούνται πληροφορίες της μορφής **-b**, όπου **b** μία πράξη που καλείται να εκτελέσει υπό συγκεκριμένες προϋποθέσεις. Ο τρόπος που εξετάζει τις πράξεις το σύστημα μέχρι στιγμής δεν εμπεριέχει πληροφορία άρνησης, παρόλο που νοηματικά είναι κατανοητό ότι η επιλογή του **move**, όπως έχει οριστεί εμπεριέχει εγγενώς την άρνηση του **park** και ανάποδα. Ο τρόπος που λύνεται αυτό είναι να εισαχθεί αυτό ακριβώς το συμπέρασμα στο σύστημα που τελικά καταλήγει ως

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move

Ακολουθεί η ανάλυση του συστήματος για κάθε περίπτωση γεγονότος.

Περίπτωση 1: **no_destination**

Αν επιλεγεί η πράξη **move** το σύστημα γίνεται

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
no_destination
move

από **move→-park** και Modus Ponens έχουμε **-park** και από **move→next_no_destination/next_destination** και Modus Ponens προκύπτει είτε **next_no_destination** είτε **next_destination**. Δε θα γίνει διαχωρισμός γι' αυτό γιατί δεν αλλάζει κάτι στην εξέταση για αντιφάσεις ή όχι, οι όροι που προκύπτουν δεν εμφανίζονται σε κάποιον κανόνα.

άρα το σύστημα είναι σε μορφή

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
no_destination

move
-park
next_no_destination/next_destination

όπου υπάρχει λειτουργική αντίφαση λόγω

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
no_destination
move
-park
next_no_destination/next_destination

και ο κόσμος απορρίπτεται

Αν επιλεγεί η πράξη **park** το σύστημα γίνεται

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
no_destination
park

από **park→-move** και Modus Ponens έχουμε **-move**

άρα το σύστημα είναι σε μορφή

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
no_destination
park
-move

που είναι αποδεκτός κόσμος.

Περίπτωση 2: **destination**

Αν επιλεγεί η πράξη **move** το σύστημα γίνεται

OB(no destination→park)

OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
destination
move

από **move→-park** και Modus Ponens έχουμε **-park** και από **move→next_no_destination/next_destination** και Modus Ponens προκύπτει είτε **next_no_destination** είτε **next_destination**.

άρα το σύστημα είναι σε μορφή

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
destination
move
-park
next_no_destination/next_destination

που είναι αποδεκτός κόσμος.

Αν επιλεγεί η πράξη **park** το σύστημα γίνεται

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
destination
park

από **park→-move** και Modus Ponens έχουμε **-move**

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
destination
park
-move

όπου υπάρχει λειτουργική αντίφαση λόγω

OB(no destination→park)

OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
destination
park
-move

άρα ο κόσμος απορρίπτεται και το σύστημα αποδέχεται τον προηγούμενο, όπου είχε επιλεγεί η πράξη **move**.

Περίπτωση 3: **new_destination**

Αν επιλεγεί η πράξη **move** το σύστημα γίνεται

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
new_destination
move

από **move→-park** και Modus Ponens έχουμε **-park** και από
move→next_no_destination/next_destination και Modus Ponens προκύπτει είτε
next_no_destination είτε **next_destination**.

άρα το σύστημα είναι σε μορφή

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
new_destination
move
-park
next_no_destination/next_destination

που είναι αποδεκτός κόσμος.

Αν επιλεγεί η πράξη **park** το σύστημα γίνεται

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
new_destination

park

από **park→-move** και Modus Ponens έχουμε **-move**

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
new_destination
park
-move

όπου υπάρχει λειτουργική αντίφαση λόγω

OB(no destination→park)
OB((destination||new_destination))→ move)
move→next_no_destination/next_destination
move→-park
park→-move
new_destination
park
-move

άρα ο κόσμος απορρίπτεται και το σύστημα αποδέχεται τον προηγούμενο, όπου είχε επιλεγεί η πράξη **move**.

Σε κάθε περίπτωση επιβεβαιώθηκε η συνέπεια αυτού του πράκτορα.

Collision processes

Το αμάξι θα περάσει σε αυτή την κατάσταση μόνο αν οι αισθητήρες του εντοπίσουν ότι υπάρχει κίνδυνος σύγκρουσης. Οι παράγοντες που θα μοντελοποιηθούν σχετικά με τις συνθήκες υπό τις οποίες υπάρχει ο κίνδυνος σύγκρουσης είναι τέσσερις και οργανωμένοι σε δύο ζεύγη.

- **C1:** Το να κινείται το αμάξι με προβληματική ταχύτητα προς κάποιο εμπόδιο (πεζό, άλλο αμάξι, φανάρι, κλπ)
- **C2:** Το να κινείται κάποιο εμπόδιο προς το αμάξι
- **C3:** Το να υπάρχει χώρος για να κινηθεί το αμάξι ώστε να αποφύγει κάποια σύγκρουση
- **C4:** Το να μην υπάρχει χώρος για να κινηθεί το αμάξι ώστε να αποφύγει κάποια σύγκρουση

Προφανώς όπως και με κάθε άλλο μοντέλο που κατασκευάζεται στα πλαίσια αυτής της εργασίας δεν αποτελεί αντικείμενο ενδιαφέροντος να αναλυθεί κάθε μία πιθανή περίπτωση. Επί παραδείγματι δεν γίνεται διαχωρισμός τι γίνεται αν και το αμάξι κατευθύνεται προς το εμπόδιο και το εμπόδιο προς το αμάξι. Η περίπτωση αυτή θεωρείται ότι εκπίπτει στο να μετακινείται το εμπόδιο προς το αμάξι καθώς ο παράγων αυτός είναι πιο επικίνδυνος από το να κινείται το αμάξι προς το εμπόδιο και γι' αυτό αντιμετωπίζεται με μεγαλύτερη βαρύτητα. Ο λόγος που θεωρείται είναι πως σε αυτήν την περίπτωση η προβληματική συμπεριφορά πηγάζει από εξωγενή παράγοντα, δηλαδή από το εμπόδιο.

Οι παράγοντες αυτοί θα αντιμετωπίζονται σαν εξωτερικά γεγονότα που έρχονται στο σύστημα.

Γεγονότα: **c1/c2** και **c3/c4**, αφού όπως εξηγήθηκε οι παράγοντες ισχύουν ανά ζεύγη.
Άρα οι πιθανές περιπτώσεις είναι τέσσερις.

c1&c3
c1&c4
c2&c3
c2&c4

Οι πράξεις, όπως και στον προηγούμενό πράκτορα εμπεριέχουν εγγενώς η μία την άρνηση της άλλης.

Οι πράξεις είναι οι εξής τρεις

- **turn:** που αφορά στο να κάνει το αμάξι μια κίνηση αποφυγής, μια στροφή ή "μανουβράρισμα" για να αποφύγει τον κίνδυνο.
- **stop:** που αφορά στο να σταματήσει.
- **manual:** που αφορά στο να δώσει το χειρισμό του αμαξιού στο χρήστη.

Ακολουθούν οι κανόνες που διέπουν τις υποχρεώσεις του συστήματος

OB((c1&c3)→turn) , δηλαδή αν το αμάξι κατευθύνεται επικίνδυνα προς κάποιο εμπόδιο και έχει τη δυνατότητα να στρίψει για να το αποφύγει, είναι υποχρεωτικό να το κάνει. Επιλέγεται να στρίψει και όχι απλά να σταματήσει για το φρενάρισμα στα μέσα του δρόμου θα αποτελεί πρόβλημα για άλλα οχήματα, πιθανώς.

OB((c1&c4)→stop) , δηλαδή αν το αμάξι κατευθύνεται επικίνδυνα προς κάποιο εμπόδιο και δεν έχει τη δυνατότητα να στρίψει για να το αποφύγει, είναι υποχρεωτικό να σταματήσει. Αν και σαφώς δεν είναι ιδανικό να σταματά στη μέση του δρόμου θεωρείται αποδεκτή λύση ανάγκης, δεδομένου ότι τα άλλα οχήματα έχουν τα συστήματα για να ανταπεξέλθουν στη στάση του αμαξιού.

OB((c2&c3)→turn) , δηλαδή αν δηλαδή αν κάποιο εμπόδιο κατευθύνεται επικίνδυνα προς το αμάξι και το αμάξι έχει τη δυνατότητα να στρίψει για να το αποφύγει, είναι υποχρεωτικό να το κάνει.

OB((c2&c4)→manual) , δηλαδή αν δηλαδή αν κάποιο εμπόδιο κατευθύνεται επικίνδυνα προς το αμάξι και το αμάξι δεν έχει τη δυνατότητα να στρίψει για να το αποφύγει, είναι υποχρεωτικό να δώσει τον έλεγχο στο χρήστη. Πρόκειται σαφώς για λύση ύστατης ανάγκης που υπάρχει απλά για να μην υπάρχει η δυνατότητα να "εγκλωβιστεί" ο οδηγός στο αμάξι παρουσία επερχόμενης σύγκρουσης και να μην μπορεί να κάνει κάτι γι' αυτό.

Το σύστημα λοιπόν είναι.

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)

για τον ίδιο λόγο που εξηγήθηκε παραπάνω επεκτείνεται με κανόνες που εκφράζουν ότι η επιλογή μιας πράξης "παράγει" την άρνηση των υπολοίπων, για να μπορεί να αξιοποιηθεί η έννοια της λειτουργικής αντίφασης ως εξής

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop

Λόγω της πανομοιότυπης συμπεριφοράς του συστήματος αν γίνει μία εκ των δύο λάθος επιλογών που πάνε αντίθετα στις υποχρεώσεις του, οι λάθος επιλογές θα παρουσιαστούν ενοποιημένα. Ωστόσο επειδή κάθε φορά μόνο το ένα θα ισχύει χρησιμοποιούμε το "//" για να δηλώσουμε ότι στην πραγματικότητα ισχύει το ένα εκ των δύο κάθε φορά.

Περίπτωση 1α: **c1&c3** και **stop//manual**

λόγω του **stop//manual** και των **stop→-turn** και **manual→-turn** και Modus Ponens παράγεται σε κάθε περίπτωση υποχρεωτικά το **-turn** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c1
c3
stop//manual
-turn

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn

stop→-manual
manual→-turn
manual→-stop
c1
c3
stop//manual
-turn

και ο κόσμος αυτός απορρίπτεται

Περίπτωση 1b: **c1&c3** και **turn**

λόγω του **turn** και των **turn→-manual** και **turn→-stop** και Modus Ponens παράγονται τα **-manual** και **-stop** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c1
c3
turn
-manual
-stop

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 2α: **c1&c4** και **turn//manual**

λόγω του **turn//manual** και των **turn→-stop** και **manual→-stop** και Modus Ponens παράγεται σε κάθε περίπτωση υποχρεωτικά το **-stop** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn

manual→-stop
c1
c4
turn//manual
-stop

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c1
c4
turn//manual
-stop

και ο κόσμος αυτός απορρίπτεται

Περίπτωση 2b: **c1&c4** και **stop**

λόγω του **stop** και των **stop→-manual** και **stop→- turn** και Modus Ponens παράγονται τα **-manual** και **- turn** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c1
c4
stop
-manual
-turn

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 3α: **c2&c3** και **stop//manual**

λόγω του **stop//manual** και των **stop→-turn** και **manual→-turn** και Modus Ponens παράγεται σε κάθε περίπτωση υποχρεωτικά το **-turn** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c2
c3
stop//manual
-turn

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c2
c3
stop//manual
-turn

και ο κόσμος αυτός απορρίπτεται

Περίπτωση 3b: **c2&c3** και **turn**

λόγω του **turn** και των **turn→-manual** και **turn→-stop** και Modus Ponens παράγονται τα **-manual** και **-stop** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn

stop→-manual
manual→-turn
manual→-stop
c2
c3
turn
-manual
-stop

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 4α: **c2&c4** και **stop//turn**

λόγω του **stop//turn** και των **turn→-manual** και **stop→-manual** και Modus Ponens παράγεται σε κάθε περίπτωση υποχρεωτικά το **-manual** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c2
c4
stop//turn
-manual

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c2
c4
stop//turn
-manual

και ο κόσμος αυτός απορρίπτεται

Περίπτωση 4b: **c2&c4** και **manual**

λόγω του **manual** και των **manual→-turn** και **manual→-stop** και Modus Ponens παράγονται τα **-turn** και **-stop** οπότε το σύστημα περνά στη μορφή

OB((c1&c3)→turn)
OB((c1&c4)→stop)
OB((c2&c3)→turn)
OB((c2&c4)→manual)
turn→-stop
turn→-manual
stop→-turn
stop→-manual
manual→-turn
manual→-stop
c2
c4
manual
-turn
-stop

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Σε κάθε περίπτωση παρουσιάστηκε η συνέπεια του πράκτορα.

Fuel processes

Το αμάξι θα περάσει σε αυτήν την κατάσταση μόνο αν το ντεπόζιτο αδειάζει και αν δεν υπάρχει κάποια διαδικασία με μεγαλύτερη προτεραιότητα η οποία να είναι απαραίτητο να χρησιμοποιηθεί. Συνεπώς, δε χρειάζεται να κάνουμε διαχωρισμό περιπτώσεων όσον αφορά σε κάποιο εξωτερικό γεγονός που να αφορά στο κατά πόσο αδειάζει το ντεπόζιτο ή όχι. Η διαδικασία ενεργοποιείται μόνον αν αδειάζει, άρα θα υπάρχει σαν πληροφορία εσωτερικά στο σύστημα της διαδικασίας.

Επίσης η πράξη που εξετάζει το σύστημα αν έχει να κάνει ή όχι θα είναι μια γενική πράξη η οποία σε επόμενο στάδιο θα ενεργοποιεί τις εσωτερικές διαδικασίες που θέλουμε να γίνουν αντί να εξετάζουμε ξεχωριστά για κάθε μια από τις λειτουργίες. Ο λόγος για αυτό είναι ότι πάντα θα θέλουμε και τις δύο λειτουργίες οπότε αφού θα συμβαίνουν μαζί, η προσέγγιση αυτή μας επιτρέπει να γλυτώνουμε τον έλεγχο περισσότερων πιθανών κόσμων.

Δεδομένα: **emptying**, που εξηγεί ότι στην παρούσα κατάσταση το ντεπόζιτο του αμαξιού είναι σχετικά κοντά στο να αδειάσει.
Εσωτερικές Λειτουργίες: **notify**, που εξηγεί ότι ο επιβάτης θα ενημερωθεί για την επερχόμενη αλλαγή πορείας προς βενζινάδικο και **seek_gas_station** , που αφορά στην αλλαγή διαδρομής ώστε να αναζητηθεί βενζινάδικο.
Πράξη: **fuel_action/-fuel_action**, που αφορά στη γενική πράξη που ελέγχει το σύστημα του τομέα αυτού αν θα εκτελέσει ή όχι.

Οι κανόνες και η ανάλυση αυτού του τμήματος είναι πρακτικά γραμμική.

OB(emptying→fuel_action) , δηλαδή είναι υποχρεωτικό εφόσον το ντεπόζιτο αδειάζει να περάσει στις δράσεις του αυτός ο τομέας,

πράγμα που γίνεται με τους κανόνες **fuel_action→ notify** (η πράξη τομέα οδηγεί άμεσα την ενημέρωση του επιβάτη) και **fuel_action→ seek_gas_station** (η πράξη τομέα οδηγεί άμεσα στην αναζήτηση βενζινάδικου).

Οπότε το σύστημα τελικά είναι

emptying
OB(emptying→fuel_action)
fuel_action→ notify
fuel_action→ seek_gas_station

Το σύστημα του τομέα, έχει θεωρητικά δύο επιλογές, να τελέσει την πράξη **fuel_action** ή να μην την τελέσει.

Περίπτωση 1: **fuel_action**

Με την πρόσθεση της πρότασης **fuel_action** το σύστημα περνά στην παρακάτω κατάσταση

emptying
OB(emptying→fuel_action)
fuel_action→ notify
fuel_action→ seek_gas_station
fuel_action

και άμεσα λόγω των κανόνων **fuel_action→ notify** και **fuel_action→ seek_gas_station** και Modus Ponens στην επόμενη κατάσταση

emptying
OB(emptying→fuel_action)
fuel_action→ notify
fuel_action→ seek_gas_station
fuel_action
notify
seek_gas_station

η οποία είναι μια αποδεκτή κατάσταση στην οποία δεν υπάρχει κάποια αντίφαση.

Περίπτωση 2: **-fuel_action**

Με την πρόσθεση της πρότασης **-fuel_action** το σύστημα περνά στην παρακάτω κατάσταση

emptying
OB(emptying→fuel_action)
fuel_action→ notify
fuel_action→ seek_gas_station

-fuel_action

όπου υπάρχει λειτουργική αντίφαση λόγω

emptying
OB(emptying→fuel_action)
fuel_action→ notify
fuel_action→ seek_gas_station
-fuel_action

άρα αυτός ο κόσμος θα απορριφθεί.

Ο πράκτορας του τομέα αυτού θα επιλέξει την πρώτη περίπτωση.
Σημειώνεται ότι όπως σε κάθε μοντέλο που παρουσιάζεται στα πλαίσια της διπλωματικής αυτής εργασίας υπάρχουν εναλλακτική τρόποι υλοποίησης του εκάστοτε τμήματος.
Παρουσιάστηκε λοιπόν η συνέπεια και του παραπάνω πράκτορα.

Start processes

Το αμάξι θα περάσει σε αυτή την κατάσταση κατά την έναρξη λειτουργίας του. Περιλαμβάνει ελέγχους για το κατά πόσο υπάρχει όντος επιβάτης και σε περίπτωση που υπάρχει, και δεν υπάρχει κάποιο πρόβλημα στο όχημα καθώς και στη θέση την οποία βρίσκεται, θα ξεκινήσει.

Γεγονότα: **safe**/**not_safe** που αφορούν στο κατά πόσο μια εσωτερική διαγνωστική διεργασία που δίνει σε αυτό τον πράκτορα πληροφορίες κρίνει ότι η ηλεκτρολογική και μηχανολογική κατάσταση του οχήματος είναι καλές και άρα το αμάξι είναι ασφαλές για χρήση ή όχι και **got_passenger**/**no_passenger** που αφορούν στο κατά πόσο υπάρχει επιβάτης ή όχι.

Πράξεις: **start**/**-start** που αφορούν στις δύο πιθανές πράξεις από τις οποίες το σύστημα μπορεί να κάνει, δηλαδή να εκκινήσει διαδικασίες για να βρεθεί σε κίνηση ή να μην τις ξεκινήσει.

Δεδομένα: **safeground/unsafeground** που αφορούν στο κατά πόσο το αμάξι θεωρεί ότι το σημείο στο οποίο βρίσκεται είναι κατάλληλο για να ξεκινήσει διαδικασίες οδήγησης ή όχι και **risk** που αφορά στο κατά πόσο το αμάξι θα έμπαινε σε κίνδυνο αν έκανε μια διαδικασία.

Εσωτερικές Λειτουργίες: **check** που αφορά στον έλεγχο που κάνει για να δει αν το σημείο στο οποίο βρίσκεται είναι κατάλληλο για εκκίνηση ή όχι και **rest** που αφορά στην μετάβαση σε μη λειτουργία.

Οι κανόνες που διέπουν το σύστημα είναι οι εξής

OB((safe & got_passenger)→start) ,δηλαδή είναι υποχρεωτικό αν υπάρχει επιβάτης και το αμάξι είναι εσωτερικά ασφαλές για χρήση, να ξεκινήσει.

OB((no_passenger||not_safe)→-start) ,δηλαδή είναι υποχρεωτικό αν δεν υπάρχει επιβάτης ή υπάρχει πρόβλημα στη λειτουργία του αμαξιού να μην ξεκινήσει.

start→check αν ξεκινήσει ελέγχει και την θέση του πριν περάσει σε διαδικασίες κίνησης.

check→safeground/unsafeground ,ο έλεγχος οδηγεί σε πληροφορία για το κατά πόσο η θέση είναι ασφαλής ή όχι.

-start→rest ,αν δεν ξεκινήσουν οι διαδικασίες κίνησης το αμάξι θα σβήσει πλήρως.

unsafeground→risk ,το μη ασφαλές έδαφος οδηγεί σε κίνδυνο αν το αμάξι επρόκειτο να κινηθεί.

risk→rest ,δηλαδή αν το αμάξι εντοπίσει κίνδυνο κατά την εκκίνηση, σταματά τη λειτουργία του.

Το τελικό σύστημα έχει ως εξής

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk→rest

Θα αναλυθούν οι διάφορες περιπτώσεις για τα γεγονότα

Υπάρχουν τέσσερις συνδυασμοί γεγονότων

Περίπτωση 1: **not_safe & got_passenger**

Αν το σύστημα επιλέξει την πράξη **start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
got_passenger
start

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest

not_safe
got_passenger
start

και ο κόσμος απορρίπτεται. Θα προέκυπταν και άλλες διαδικασίες, λόγω **start→check** αλλά επειδή είναι προφανές εξ αρχής ότι υπάρχει λειτουργική αντίφαση, ο έλεγχος σταματά εδώ.

Αν το σύστημα επιλέξει την πράξη **-start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
got_passenger
-start

και έπειτα λόγω **-start→rest** και Modus Ponens στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
got_passenger
-start
rest

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 2: **not_safe & no_passenger**

Αν το σύστημα επιλέξει την πράξη **start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
no_passenger

start

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
no_passenger
start

και ο κόσμος απορρίπτεται. Θα προέκυπταν και άλλες διαδικασίες, λόγω **start→check** αλλά επειδή είναι προφανές εξ αρχής ότι υπάρχει λειτουργική αντίφαση, ο έλεγχος σταματά εδώ.

Αν το σύστημα επιλέξει την πράξη **-start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
no_passenger
-start

και έπειτα λόγω **-start→rest** και Modus Ponens στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
not_safe
no_passenger
-start
rest

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 3: **safe & no_passenger**

Αν το σύστημα επιλέξει την πράξη **start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
no_passenger
start

όπου υπάρχει λειτουργική αντίφαση λόγω

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
no_passenger
start

και ο κόσμος απορρίπτεται. Θα προέκυπταν και άλλες διαδικασίες, λόγω **start→check** αλλά επειδή είναι προφανές εξ αρχής ότι υπάρχει λειτουργική αντίφαση, ο έλεγχος σταματά εδώ.

Αν το σύστημα επιλέξει την πράξη **-start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
no_passenger
-start

και έπειτα λόγω **-start→rest** και Modus Ponens στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest

unsafeground→risk
risk->rest
safe
no_passenger
-start
rest

όπου δεν υπάρχει αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 4: **safe & got_passenger**

Αν το σύστημα επιλέξει την πράξη **-start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
-start

όπου υπάρχει λογική αντίφαση λόγω

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
-start

και ο κόσμος απορρίπτεται.

Αν το σύστημα επιλέξει την πράξη **start** περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe

got_passenger
start

και λόγω **start→check** και Modus Ponens στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
start
check

σε αυτό το σημείο εξετάζονται δύο διαφορετικές υποπεριπτώσεις λόγω
check→safeground/unsafeground και Modus Ponens

Περίπτωση 4α: **safe & got_passenger & safeground** το σύστημα περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
start
check
safeground

όπου δεν υπάρχει καμία αντίφαση και ο κόσμος γίνεται αποδεκτός.

Περίπτωση 4β: **safe & got_passenger & unsafeground** το σύστημα περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
start

check
unsafeground

και λόγω **unsafeground→risk** και Modus Ponens το σύστημα περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
start
check
unsafeground
risk

και λόγω **risk->rest** και Modus Ponens το σύστημα περνά στην κατάσταση

OB((safe & got_passenger)→start)
OB((no_passenger||not_safe)→-start)
start→check
check→safeground/unsafeground
-start→rest
unsafeground→risk
risk->rest
safe
got_passenger
start
check
unsafeground
risk
rest

όπου δεν υπάρχει καμία αντίφαση και ο κόσμος γίνεται αποδεκτός.

Παρουσιάστηκε η συνέπεια και αυτού του πράκτορα σε κάθε περίπτωση.

Supervisor

Στο τελευταίο στάδιο θα παρουσιαστεί η λειτουργία του πράκτορα που επιβλέπει τους παραπάνω και επιλέγει ποιος από αυτούς θα είναι σε ισχύ κάθε φορά. Η λειτουργία του είναι ιδιαίτερα διαφορετική από αυτή του επιβλέποντα στο τελευταίο ιατρικό μοντέλο. Εκείνος μετέδιδε πληροφορία μεταξύ των επιμέρους πρακτόρων, ενώ αυτός απλώς "καλεί" μηχανικά τον ένα ή τον άλλον ανάλογα με τις πληροφορίες που έχει. Μοντελοποιείται απλώς η συμπεριφορά του μια δεδομένη στιγμή χωρίς να εξετάζεται πως γίνεται η ανανέωση του συστήματός του από τη μία

στιγμή στην επόμενη.

Ο επιβλέπων λειτουργεί ως εξής. Τέσσερα πιθανά γεγονότα μπορεί να ισχύουν ή να μην ισχύουν. Τα γεγονότα αυτά αντιπροσωπεύουν ότι ο επιβλέπων εντόπισε μια συνθήκη σχετική με τη λειτουργία του κάθε πράκτορα, χωρίς να μπαίνουμε σε λεπτομέρειες ως προς το τι ακριβώς συμβαίνει, καθώς αυτό είναι κάτι που το επιλύει ο κάθε επιμέρους πράκτορας με τους δικούς του αισθητήρες. Ανάλογα με το πιο γεγονός ισχύει ενεργοποιεί τον αντίστοιχο πράκτορα. Αν ισχύουν παραπάνω από ένα, επιλέγει με βάση μια ιεραρχία που ορίζεται παρακάτω.

1). Αν δεν ενεργοποιηθούν οι διαδικασίες κίνησης, οι υπόλοιπες ούτως ή άλλως δεν πρόκειται να λειτουργούν. Άρα το **Start processes** είναι πρώτο στην ιεραρχία.
2). Πρέπει να αποφεύγονται συγκρούσεις αν είναι δυνατόν, αν δηλαδή μπορεί να κινηθεί το αμάξι. Άρα το **Collision processes** είναι δεύτερο στην ιεραρχία.
3). Πρέπει να επιλέγει να γεμίζει τα καύσιμα, αν δεν προέχει κίνδυνος σύγκρουσης και το ντεπόζιτο κινδυνεύει να αδειάσει. Άρα το **Fuel processes** είναι τρίτο στην ιεραρχία.
4)Σε περίπτωση που δεν υπάρχει κάποιος κίνδυνος και το αμάξι είναι σε κίνηση, προχωρά στη λειτουργία διεκπεραίωσης διαδρομών. Άρα το **Destination processes** είναι τέταρτο στην ιεραρχία.

Οι όροι άδειας για τον κάθε πράκτορα είναι **p1**, **p2** ,**p3** ,**p4** και οι όροι ανάγκης **n1**, **n2**, **n3**, **n4** αντίστοιχα. Η ιεραρχία χτίζεται βεβαιώνοντας ότι η κάθε πράξη επιλέγεται μόνο αν υπάρχει ανάγκη για αυτήν και δεν υπάρχει ανάγκη για κάποια μεγαλύτερης προτεραιότητας.
Πέραν από τους κανόνες που βεβαιώνουν την ιεραρχία υπάρχουν επιπλέον κανόνες που "παράγουν" την άρνηση των πράξεων που δεν επιλέχτηκαν για να αξιοποιηθεί κατάλληλα η έννοια της λειτουργικής αντίφασης.

Κανόνες ιεραρχίας

OB(n1→p1)
OB((n2&-n1)→p2))
OB((n3&-n2&-n1)→p3)
OB((n4&-n3&-n2&-n1)→p4)

Κανόνες δημιουργίας άρνησης

p1→-p2
p1→-p3
p1→-p4
p2→-p1
p2→-p3
p2→-p4
p3→-p1
p3→-p2
p3→-p4
p4→-p1
p4→-p2
p4→-p3

Λόγω του μεγάλου πλήθους λειτουργικά πανομοιότυπων περιπτώσεων η εξέταση του συστήματος θα γίνει συνοπτικά

Αν έχουμε ανάγκη τύπου **n1** ανεξάρτητα από το τι άλλες ανάγκες έχουμε υπάρχουν δύο εκδοχές
α) να επιλεγεί μία εκ των **p2,p3,p4**
β) να επιλεγεί η **p1**

Στην επιλογή α) λόγω οποιουδήποτε εκ των

p2→-p1
p3→-p1
p4→-p1

και Modus Ponens θα "παραχθεί" το **-p1**

οπότε λόγω **OB(n1→p1), n1** και **-p1** θα υπάρχει λειτουργική αντίφαση και ο κόσμος θα απορριφθεί

Στην επιλογή β) δεν πρόκειται να έχουμε λογική αντίφαση αφού αφενός ο κανόνας **OB(n1→p1)** ικανοποιείται πλήρως και αφετέρου οι κανόνες

OB((n2&-n1)→p2))
OB((n3&-n2&-n1)→p3)
OB((n4&-n3&-n2&-n1)→p4)

δεν πρόκειται να προκαλέσουν αντίφαση γιατί το αριστερό μέλος δε θα εκπληρώνεται ποτέ λόγω **n1**.

Άρα η επιλογή β) προκαλεί αποδεκτό κόσμο και επιλέγεται.

Αν έχουμε ανάγκη τύπου **n2** και δεν έχουμε ανάγκη **n1,** δηλαδή έχουμε **-n1** (αν είχαμε **n1** παρουσιάστηκε τι ισχύει παραπάνω) ανεξάρτητα από το τι άλλες ανάγκες έχουμε υπάρχουν δύο εκδοχές
α) να επιλεγεί μία εκ των **p1,p3,p4**
β) να επιλεγεί η **p2**

Στην επιλογή α) λόγω οποιουδήποτε εκ των

p1→-p2
p3→-p2
p4→-p2

και Modus Ponens θα "παραχθεί" το **-p2**

οπότε λόγω **OB((n2&-n1)→p2)), n2, -n1** και **-p2** θα υπάρχει λειτουργική αντίφαση και ο κόσμος θα απορριφθεί

Στην επιλογή β) δεν πρόκειται να έχουμε λογική αντίφαση αφού αφενός ο κανόνας **OB((n2&-n1)→p2))** ικανοποιείται πλήρως και αφετέρου οι κανόνες

OB(n1→p1)
OB((n3&-n2&-n1)→p3)
OB((n4&-n3&-n2&-n1)→p4)

δεν πρόκειται να προκαλέσουν αντίφαση γιατί το αριστερό μέλος δε θα εκπληρώνεται ποτέ λόγω **-n1** και **n2**.

Άρα η επιλογή β) προκαλεί αποδεκτό κόσμο και επιλέγεται.

Αν έχουμε ανάγκη τύπου **n3** και δεν έχουμε ανάγκη **n1** ή **n2** δηλαδή έχουμε **-n1** και **-n2** (αν είχαμε **n1** ή **n2** παρουσιάστηκε τι ισχύει παραπάνω) ανεξάρτητα από το τι γίνεται με το **n4** υπάρχουν δύο εκδοχές
α) να επιλεγεί μία εκ των **p1,p2,p4**
β) να επιλεγεί η **p3**

Στην επιλογή α) λόγω οποιουδήποτε εκ των

p1→-p3
p2→-p3
p4→-p3

και Modus Ponens θα "παραχθεί" το **-p3**

οπότε λόγω **OB((n3&-n2&-n1)→p3), n3, -n1 , -n2** και **-p3** θα υπάρχει λειτουργική αντίφαση και ο κόσμος θα απορριφθεί

Στην επιλογή β) δεν πρόκειται να έχουμε λογική αντίφαση αφού αφενός ο κανόνας **OB((n3&-n2&-n1)→p3)** ικανοποιείται πλήρως και αφετέρου οι κανόνες

OB(n1→p1)
OB((n2&-n1)→p2))
OB((n4&-n3&-n2&-n1)→p4)

δεν πρόκειται να προκαλέσουν αντίφαση γιατί το αριστερό μέλος δε θα εκπληρώνεται ποτέ λόγω **-n1, -n2** και **n3**.

Άρα η επιλογή β) προκαλεί αποδεκτό κόσμο και επιλέγεται.

Αν έχουμε ανάγκη τύπου **n4** και δεν έχουμε ανάγκη **n1** ή **n2** ή **n3** δηλαδή έχουμε **-n1** και **-n2** και **-n3** (αν είχαμε **n1** ή **n2** ή **n3** παρουσιάστηκε τι ισχύει παραπάνω) υπάρχουν δύο εκδοχές
α) να επιλεγεί μία εκ των **p1,p2,p3**
β) να επιλεγεί η **p4**

Στην επιλογή α) λόγω οποιουδήποτε εκ των

p1→-p4
p2→-p4

p3→-p4

και Modus Ponens θα "παραχθεί" το **-p4**

οπότε λόγω **OB((n4&-n3&-n2&-n1)→p4), n4, -n1 , -n2 , -n3** και **-p4** θα υπάρχει λειτουργική αντίφαση και ο κόσμος θα απορριφθεί

Στην επιλογή β) δεν πρόκειται να έχουμε λογική αντίφαση αφού αφενός ο κανόνας **OB((n4&-n3&-n2&-n1)→p4)** ικανοποιείται πλήρως και αφετέρου οι κανόνες

OB(n1→p1)
OB((n2&-n1)→p2))
OB((n3&-n2&-n1)→p3)

δεν πρόκειται να προκαλέσουν αντίφαση γιατί το αριστερό μέλος δε θα εκπληρώνεται ποτέ λόγω **-n1, -n2** και **-n3**.

Άρα η επιλογή β) προκαλεί αποδεκτό κόσμο και επιλέγεται.

Σε κάθε περίπτωση παρουσιάστηκε η συνέπεια και του επιβλέποντος πράκτορα.

Κεφάλαιο 4

Συμπεράσματα

Το βασικότερο συμπέρασμα που εξήγαγα στα πλαίσια αυτής της διπλωματικής εργασίας είναι ότι η αξιοποίηση της δεοντικής λογικής για επιτυχή ανάπτυξη ενός συστήματος που να αποσκοπεί στην επίτευξη της κατασκευής ενός μοντέλου συμπεριφοράς πρακτόρων τεχνητής νοημοσύνης είναι μια ιδέα που σίγουρα έχει προοπτικές. Τόσο σε επίπεδο συνέπειας όσο και στο πόσο άμεσα ξεκάθαρη είναι η ορθότητα των συστημάτων βασισμένων στη deontic logic, τα μοντέλα που αναπτύχθηκαν ήταν επιτυχή.
Αυτό σίγουρα ανοίγει προοπτικές για περαιτέρω έρευνα σε αυτόν τον τομέα, ο οποίος είναι ιδιαίτερα νέος αλλά και σημαντικός για το μέλλον δεδομένου της συνεχώς αυξανόμενης συμμετοχής συστημάτων τεχνητής νοημοσύνης στη ζωή μας.
Επίσης, φάνηκε ξεκάθαρα η σημασία της προσαρμογής των μοντέλων στις διαφορετικές περιπτώσεις καθώς με αυτόν τον τρόπο επιτυγχάνεται μέγιστη δυνατή ανταπόκριση στα προβλήματα που μοντελοποιούνται κάθε φορά.
Τέλος, το πιο σημαντικό ίσως συμπέρασμα αυτής της διπλωματικής εργασίας, όπως φάνηκε σε κάθε στάδιο ανάπτυξης των μοντέλων που παρουσιάστηκαν, είναι πως οι μεθοδολογίες που θα πρέπει να αναπτύσσονται, θα πρέπει να εξελίσσονται συνεχώς και να εφοδιάζονται με περαιτέρω εργαλεία ώστε να μπορούν να ανταποκρίνονται στις περαιτέρω ανάγκες που προκύπτουν όσο γίνεται πιο πολύπλοκο το πρόβλημα που μοντελοποιείται. Η ανάπτυξη συστημάτων και μοντέλων τεχνητής νοημοσύνης με εξασφαλισμένα ορθή ηθική συμπεριφορά είναι μια όχι στατική, αλλά δυναμική διαδικασία.

Παράρτημα Α’

Κώδικες Επίδειξης

Ακολουθούν κώδικες προγραμμάτων σε γλώσσα C που εξομοιώνουν οπτικά τη συμπεριφορά ενός ρομπότ εφοδιασμένου με τα συστήματα των τριών πρώτων ιατρικών μοντέλων. Σημειώνεται ότι η συμπεριφορά παρουσιάζεται οπτικά, σε επίπεδο χρήστη, χωρίς να υπάρχει εσωτερικά πλήρης αναπαράσταση του συστήματος.

Αναπτύχθηκαν σε λειτουργικό περιβάλλον Dev-C++, που δε διαθέτει ελληνικούς χαρακτήρες.

medical1.c

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>


int main () {

	int  x;
	printf("If patient asks for medicine, the robot has to give them medicine \n");
	printf("Press 1 if patient asks for medicine \n");
		scanf("%d",&x);

	if(x==1){
	printf("Patient is given the medicine \n");
	};
	system("PAUSE");
}
```

medical2.c

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>


int main () {

	int  x;
	printf("If patient asks for medicine, the robot has to give them medicine \n");
	printf("\n");
```

```
        printf("If patient doesn't get medicine, they might die \n");
        printf("The robot has to keep the patient from dying \n");
        printf("\n");

        printf("Press 1 if patient asks for medicine \n");
        printf("Press any key otherwise \n");

        scanf("%d",&x);

        if(x==1){
        printf("Patient is given the medicine \n");
        }
        else {printf("Patient is given the medicine anyway to keep them from dying \n");

        }
        system("PAUSE");

}
```

medical3.c

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>


int main () {

        int  x;
        int flag=0;
        printf("If patient asks for medicine, the robot has to give them medicine \n");
        printf("If patient doesn't ask for medicine, the robot has to respect their choice \n");
        printf("\n");

        printf("If patient doesn't get medicine, they might die \n");
        printf("The robot has to keep the patient from dying \n");
        printf("\n");

        printf("Press 1 if patient asks for medicine \n");
        printf("Press any key otherwise \n");

        scanf("%d",&x);

        if(x==1){
        printf("Patient is given the medicine \n");
        }
        else {printf("Patient is not given the medicine \n");
                flag=1;
```

```
	}
	if (flag==1){
			printf("\n");
		printf("The situation exceeds the robots capabilities \n");
			printf("CALLING SUPERVISOR \n");


	}
	system("PAUSE");

}
```

medical4.c

```
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>


int main () {

	int  x;
	int y;
	int flag=0;

	int num;

	time_t t1;
	srand((unsigned) time(&t1));


	printf("The system first calculates the course of action for maximum happiness,\n");
	printf("under the following conditions: \n");
	printf("If patient asks for medicine, the robot has to give them medicine \n");
	printf("If patient doesn't ask for medicine, the robot has to respect their choice \n");
	printf("\n");

	printf("Press 1 if patient asks for medicine \n");
	printf("Press any key otherwise \n");
	scanf("%d",&x);

	printf("Press 1 if there is a large stock \n");
	printf("Press any key otherwise \n");
	scanf("%d",&y);
	printf("\n");

	printf("The following conditions are also considered in the main system:\n");
	printf("If patient doesn't get medicine, the robot checks their condition \n");
	printf("If patient doesn't get medicine and is in a bad condition, they might die \n");
```

```
        printf("The robot has to keep the patient from dying \n");

        printf("\n");


        if(x==1){
        printf("Patient is given the medicine\n");
        }
        else {printf("Patient is not given the medicine \n");
                printf("Checking patient's condition as obligated \n");
                //patient's condition is generated randomly
                num=rand()%2;
                if(num==1){
                printf("Patient is relatively okay and can survive the day without medicine \n");
                }
                else{
                                printf("Patient is in critical condition \n");
                                        flag=1;

                }

        }
        if (flag==1){
                        printf("\n");
                        printf("The situation exceeds the robots capabilities \n");
                        printf("CALLING SUPERVISOR \n");}
        system("PAUSE");

}
```

Βιβλιογραφία

- **[1]** https://plato.stanford.edu/entries/logic-deontic/
- **[2]** https://plato.stanford.edu/entries/logic-modal/
- **[3]** I. Asimov - Runaround, Street & Smith, 1942