ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

Διπλωματική Εργασία

«Ένα υπολογιστικό περιβάλλον για τη θεωρία των Κανονιστικών Θέσεων»

«A computational framework for the theory of Normative Positions»

Μούτουπας Κων/νος

Επιβλέπουσα Καθηγήτρια: Δασκαλοπούλου Ασπασία

Επίκουρη Καθηγήτρια Π.Θ

Δεύτερο Μέλος Επιτροπής: Τσουκαλάς Ελευθέριος

Καθηγητής Π.Θ

Βόλος, 2016

Ευχαριστίες,

Ευχαριστώ θερμά την κυρία Δασκαλοπούλου Ασπασία για την μεγάλη υποστήριξη και καθοδήγηση που μου έδωσε κατά τη διάρκεια της διπλωματικής μου εργασίας.

Επίσης, θα ήθελα να ευχαριστήσω τον κύριο Τσουκαλά Ελευθέριο που δέχτηκε να είναι επιβλέπων της εργασίας μου.

Τέλος, θα ήθελα να ευχαριστήσω την οικογένεια μου που με στηρίζει σε κάθε μου βήμα.

ΜΟΥΤΟΥΠΑΣ ΚΩΝ/ΝΟΣ

ΒΟΛΟΣ, 2016

Περιεχόμενα

Πίνακας εικόνων

ΚΕΦΑΛΑΙΟ 1 – ΕΙΣΑΓΩΓΗ

Η παρούσα διπλωματική ασχολείται με τη θεωρία των Κανονιστικών Θέσεων(Normative Positions) και παρουσιάζει μια πλατφόρμα που αυτοματοποιεί τη διαδικασία παραγωγής τους. Η θεωρία αυτή μελετάει τη σχέση ανάμεσα σε ένα πράκτορα και μια κατάσταση, περιγράφοντας τη δεοντική τοποθέτηση του πράκτορα σε σχέση με αυτή τη κατάσταση. Συγκεκριμένα, μελετάει αν ο πράκτορας επιτρέπεται/ απαγορεύεται/ υποχρεούται να κάνει κάτι ώστε να επέλθει αυτή η κατάσταση.

Αρχικά, έγινε μια προσπάθεια να εφαρμοστούν τα θεωρήματα που εισήγαγε ο W.N. Hohfeld [Hohfeld, 1913] , στο τομέα της Νομικής Θεωρίας, πάνω στη λήψη αποφάσεων για φυσικούς πράκτορες, οι οποίοι πρέπει με κάποιο τρόπο να συνεργαστούν ή να καταστρώσουν μαζί κάποιο σχέδιο για να επιτύχουν το στόχο τους. Ό πρώτος που προσπάθησε να το καταφέρει αυτό ήταν ο S. Kanger, ο οποίος ανάλυσε τις 4 έννοιες που δημιούργησε ο W.N. Hohfeld και από αυτές παρήγαγε 26 διαφορετικές έννοιες που περιγράφουν δεσμούς που μπορούν να αναπτυχθούν ανάμεσα σε 2 φυσικούς πράκτορες.[Kanger and Kanger, 1966][Kanger, 1971][Kanger, 1972] [Kanger, 1985]

Αργότερα, διάφοροι επιστήμονες με κυριότερο τον L. Lindahl, παρατήρησαν ότι, οι ιδέες του S. Kanger μπορούν να εφαρμοστούν σε οποιαδήποτε τομέα μπορούν να εφαρμοστούν όροι όπως “άδεια” ή “νόμιμο” με χαρακτηριστικά παραδείγματα, τα υπολογιστικά συστήματα ή τα πολυπρακτορικά συστήματα, όπου 2 ή περισσότεροι πράκτορες συνεργάζονται προκειμένου να επιτύχουν κάποιο στόχο. Έτσι, πήραν αυτή την ιδέα, την ανέπτυξαν και την τελειοποίησαν.

Ο L. Lindahl, μέσα από τη δική του ανάλυση, κατάφερε να δημιουργήσει 35 διαφορετικές έννοιες που ταιριάζουν στους δεσμούς που περιγράφει ο S. Kanger και 127 έννοιες αν είναι γνωστό ποιος είναι ο συγκεκριμένος σκοπός που θέλουν να εκπληρώσουν οι πράκτορες. Οι έννοιες αυτές μπορούν να αυξηθούν ακόμα περισσότερο αν βελτιστοποιηθούν και αν προστεθούν περισσότεροι από 2 πράκτορες. Παρόμοιες τεχνικές εφάρμοσε και ο I. Porn πάνω στη θεωρία δράσης.[Lindahl, 1977] [Porn, 1977] Ο M. Sergot βασίστηκε στις θεωρία των Kanger-Lindahl για να βελτιστοποιήσει και για να γενικεύσει τις έννοιες που εισήγαγαν έτσι ώστε να τις εφαρμόσει σε περισσότερους από 2 πράκτορες. Τέλος, έδωσε ιδέες και τρόπους για να αυτοματοποιηθεί η εφαρμογή τους σε περισσότερους από 2 πράκτορες, καθώς όσο αυξάνεται ο αριθμός των πρακτόρων ο υπολογισμός των Κανονιστικών Θέσεων με τον παραδοσιακό τρόπο τείνει να γίνεται ολοένα και πιο επίπονος.[Sergot, 2001]

1.1 Πολυπρακτορικά συστήματα

Ο M. Wooldridge το 2002 ανέφερε ότι "δεν υπάρχει ένα μοναδικό και ενιαίο σύστημα για πράκτορες". Ένα πολυπρακτορικό σύστημα είναι ένα σύστημα το οποίο αποτελείται από πράκτορες μέσα σε ένα περιβάλλον. Οι πράκτορες αυτοί μπορούν να συνεργαστούν για να λύσουν προβλήματα τα οποία είναι δύσκολο ή και ακατόρθωτο να λυθούν από ένα πράκτορα. Ένα από τα βασικά προβλήματα των πολυπρακτορικών συστημάτων είναι οι δεσμοί που πρέπει να αναπτύξουν οι πράκτορες μεταξύ τους προκειμένου να γνωρίζουν κάθε στιγμή τι άδειες έχουν, προκειμένου να είναι σε θέση να πάρουν τη καλύτερη δυνατή απόφαση για να επιτύχουν το σκοπό τους.

Τώρα είναι εύκολο λοιπόν να καταλάβει κανείς γιατί ήταν τόσο σημαντικό να εφαρμοστούν οι έννοιες του W.N. Hohfeld στα πολυπρακτορικά συστήματα. Οι θεωρίες των Kanger-Lindahl-Porn αναφέρονται στους δεσμούς που μπορούν να αναπτυχθούν ανάμεσα σε 2 πράκτορες που ακολουθούν το πρότυπο "ο πράκτορας x θα φροντίσει να γίνει το F", όπου x είναι το όνομα του πράκτορα και F η πράξη που καλείται να εκπληρώσει.

Στα πολυπρακτορικά συστήματα, ένας πράκτορας καλείται από έναν άλλο πράκτορα για να εκτελέσει κάποια πράξη. Έτσι δημιουργείται μεταξύ τους ένας δεσμός που θα μπορούσε να χαρακτηριστεί ως "υποχρέωση" ή "καθήκον". Πάνω σε αυτούς τους δεσμούς βρίσκουν εφαρμογή οι ιδέες του W.N. Hohfeld, καθώς ίδιοι δεσμοί δημιουργούνται και στη καθημερινή ζωή όταν 2 άτομα καλούνται να συνεργαστούν για να επιτύχουν κάποιο στόχο.

Έτσι, οι Κανονιστικές Θέσεις χρησιμοποιούνται για να διασαφηνιστούν οι δεοντικές σχέσεις μεταξύ των πρακτόρων. Επίσης, βρίσκουν ευρύτερη εφαρμογή στον τυπικό προσδιορισμό(formal specification) λογισμικού προκειμένου να διασαφηνιστεί ο ρόλος που παίζει το κάθε υπολογιστικό συστατικό σε όλο το σύστημα.[Jones and Sergot, 1993]

1.2 Κίνητρα για τη δημιουργία πλατφόρμας

Σε ένα πολυπρακτορικό σύστημα, οι πράκτορες που συνεργάζονται μεταξύ τους είναι συνήθως περισσότεροι από 2. Έτσι, ο M. Sergot ανέπτυξε τη θεωρία των Kanger-Lindahl-Porn και την εφάρμοσε σε περισσότερους από 2 πράκτορες. Συγκεκριμένα:

- Η θεωρία του μπορεί να εφαρμοστεί σε έναν οποιαδήποτε αριθμό πρακτόρων καθώς και σε περιπτώσεις όπου ο αριθμός των πρακτόρων δεν είναι γνωστός.
- Σε αντίθεση με την θεωρία των Kanger-Lindahl-Porn, η θεωρία του M. Sergot επιτρέπει έναν οποιαδήποτε αριθμό από θέσεις καθώς και τον κάθε δυνατό συνδυασμό τους.
- Ο υπολογισμός των θέσεων δε βασίζεται σε υποθέσεις σχετικά με τη λογική που θα χρησιμοποιηθεί. Αυτό βοηθάει στη παραγωγή πιο περίπλοκων θέσεων.
- Η θεωρία του μπορεί να εφαρμοστεί σε πιο πρακτικό επίπεδο, κάτι που βοηθάει στο καλύτερο εντοπισμό των προβλημάτων και στην πιο λεπτομερή ανάλυσή τους.
- Η θεωρία του μπορεί να αυτοματοποιηθεί.

Σκοπός, λοιπόν, αυτής της διπλωματικής εργασίας είναι, να χρησιμοποιηθεί η θεωρία που ανέπτυξε ο M. Sergot προκειμένου να αυτοματοποιηθεί ο τρόπος που υπολογίζονται οι σχέσεις μεταξύ των πρακτόρων. Συγκεκριμένα, θα δημιουργηθεί μια πλατφόρμα με τη χρήση της γλώσσας προγραμματισμού Java, η οποία θα παράγει όλες τις Κανονιστικές Θέσεις που μπορούν να αναπτυχθούν μεταξύ των πρακτόρων.

Η ανάγκη για την ύπαρξη μιας τέτοιας πλατφόρμας είναι σημαντική καθώς όσο αυξάνεται ο αριθμός των πρακτόρων, αυξάνονται ταυτόχρονα και οι πιθανές Κανονιστικές Θέσεις που μπορούν να αναπτυχθούν μεταξύ τους, κάτι που μπορεί να κάνει τη διαδικασία του υπολογισμού τους αργή, μεγάλη και αρκετά επίπονη.

1.3 Δομή της εργασίας

Η παρούσα εργασία εστιάζει στο πώς θα δημιουργηθεί αυτή η πλατφόρμα. Στο 2^{ο} κεφάλαιο αυτής της εργασίας παρουσιάζεται εκτενώς το θεωρητικό υπόβαθρο που χρειάζεται να κατέχει κάποιος για να καταλάβει τον τρόπο υπολογισμού των Normative θέσεων.

Στο 3^{ο} κεφάλαιο παρουσιάζεται ο αλγόριθμος που απαιτείται για τον υπολογισμό των Normative θέσεων καθώς και λεπτομερείς επεξηγήσεις για τον τρόπο λειτουργίας του.

Στο 4^{ο} κεφάλαιο υπάρχουν παραδείγματα πάνω στα οποία εκτελείται ο αλγόριθμος καθώς και τα συμπεράσματα που προκύπτουν από αυτά.

Τέλος, στο 5^{ο} κεφάλαιο παραθέτονται τυχόν βελτιώσεις που μπορούν να γίνουν στον αλγόριθμο όπως επίσης και ενδεχόμενες αλλαγές ή προσθήκες που ίσως απαιτηθεί να πραγματοποιηθούν μετά τη διεκπεραίωση της εργασίας.

ΚΕΦΑΛΑΙΟ 2 – NORMATIVE ΘΕΣΕΙΣ

2.1 Fundamental legal conceptions

Ο W.N. Hohfeld παρατήρησε ότι πολλοί νομικοί δεν είχαν καταλάβει σωστά την έννοια του όρου “δικαίωμα”, κάτι που τους οδηγούσε να αλλάζουν πολλές φορές το νόημα του όρου μέσα σε κάθε πρόταση. Όπως φαίνεται και στο σχήμα 1, ο W.N. Hohfeld χώρισε την έννοια του όρου “άδεια” σε 2 ομάδες από 4 έννοιες η καθεμία, οι οποίες έχουν αναπτύξει μεταξύ τους διάφορες σχέσεις. Μάλιστα, ο ίδιος τις ονόμασε “Fundamental legal conceptions”. Για παράδειγμα, οι έννοιες δικαίωμα(right) και υποχρέωση(duty) συσχετίζονται μεταξύ τους καθώς αν ένας πράκτορας x έχει κάποιο δικαίωμα πάνω σε έναν άλλο πράκτορα y που να αναγκάζει τον πράκτορα y να εκτελέσει κάποια ενέργεια F, τότε αυτό ταυτόχρονα σημαίνει ότι ο πράκτορας y έχει υποχρέωση στον πράκτορα x ότι πρέπει να εκτελέσει κάποια ενέργεια F και αντίστροφα. Αυτές τις σχέσεις, ο L. Lindahl στη βιβλιογραφία του τις εκφράζει me συντομογραφία ως εξής[Lindahl, 1977] :

Right(x, y, F) ↔ Duty (y, x, F)

Right(x, y, not-F) ↔ Duty (y, x, not-F)

Με τον όρο not-F να υποδηλώνει ότι ο πράκτορας δε θα εκτελέσει την ενέργεια F.

Αντίθετα, οι έννοιες duty(υποχρέωση) και privilege(προνόμιο) είναι αντίθετες καθώς, ένας πράκτορας x έχει προνόμιο πάνω σε ένα πράκτορα y που να αναγκάζει τον πράκτορα y να εκτελέσει κάποια ενέργεια F, όταν ο πράκτορας x δε χρωστάει κάποια υποχρέωση στον πράκτορα y που να αναγκάζει τον πράκτορα y να μην εκτελέσει την ενέργεια F και αντίστροφα. Αυτές οι σχέσεις, σε συντομογραφία εκφράζονται ως εξής:

Privilege(x, y, F) $\leftrightarrow$ ¬Duty (x, y, not-F)

Privilege(x, y, not-F) $\leftrightarrow$ ¬Duty (x, y, F)

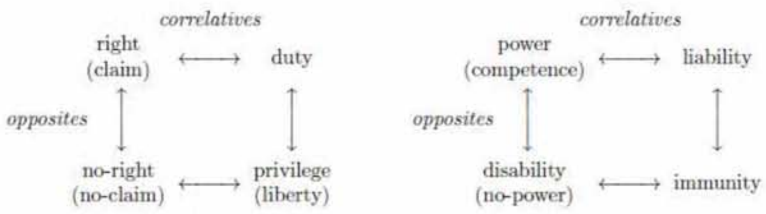
Αντίστοιχα, οι έννοιες no-right / privilege και right / no-right είναι και αντίστροφες και συσχετίζονται και σε συντομογραφία εκφράζονται ως εξής:

Right(x, y, F) $\leftrightarrow$ ¬No-right(x, y, F)

Right(x, y, not-F) $\leftrightarrow$ ¬No-right(x, y, not-F)

No-right(x, y, F) $\leftrightarrow$ Privilege(y, x, not-F)

No-right(x, y, not-F) $\leftrightarrow$ Privilege(y, x, F)



Σχήμα 1: Hohfeld's `fundamental legal conceptions', M. Sergot, 2001, page 4, Figure 1

Όπως παρατήρησαν όμως πολύ εύστοχα οι Kanger και L. Lindahl, οι θέσεις right / duty, no-right / privilege, right / no-right και duty / privilege, αποτελούνται από έννοιες οι οποίες δεν είναι του ίδιου τύπου.[Kanger and Kanger, 1966] Αυτό δημιούργησε εμπόδια στο να αναπτυχθεί μια ακριβής θεωρία που να μπορεί να εφαρμοστεί στο τομέα των πολυπρακτορικών συστημάτων. Ωστόσο, ο S. Kanger, βασιζόμενος στον τρόπο σκέψης και τις ιδέες του W.N. Hohfeld, κατάφερε να αναπτύξει τη δική του θεωρία. Αργότερα, ο L. Lindahl κατάφερε να αναπτύξει και να βελτιστοποιήσει τη θεωρία του S. Kanger και έτσι δημιουργήθηκε η θεωρία των Κανονιστικών Θέσεων.

2.2 Εισαγωγή στη θεωρία των Kanger-Lindahl

Ο S. Kanger, για να αναπτύξει τη δικιά του θεωρία χρησιμοποίησε τα βασικά εργαλεία της τυπικής λογικής(modal logic). Έτσι, η θεωρία των Kanger-Lindahl αποτελείται από ένα σύστημα με 3 τελεστές(operators): *Shall* και *May* για να περιγράψει το τύπο της σχέσης ανάμεσα στους πράκτορες και *Do* για να περιγράψει την ενέργεια που πρέπει να εκτελέσουν. Αργότερα, ο I. Porn άλλαξε τα ονόματα αυτών των τελεστών, για καλύτερη συντομογραφία και για να μπορούν να τροποποιηθούν μέσα στις εκφράσεις πιο εύκολα. Έτσι, στις θέσεις των Shall και May, μπήκαν οι O(obligation) και P(permission) και ο τελεστής Do αντικαταστάθηκε με τον E_x όπου x το όνομα του πράκτορα που εκτελεί την ενέργεια. Οι τελεστές O και P είναι αντίστροφοι, δηλαδή $O = \neg P$.

Το συγκεκριμένο σύστημα ονομάστηκε "Standard Deontic Logic"(SDL) και αποτελείται από τους παρακάτω κανόνες[Chellas, 1980] :

O.RE $(A \leftrightarrow B) / (OA \leftrightarrow OB)$

O.M $\quad O(A \land B) \rightarrow (OA \land OB)$

O.C $\quad (OA \land OB) \rightarrow O(A \land B)$

O.P $\quad \neg O\bot$

Από τους κανόνες O.C και O.P παράγεται το αξίωμα:

O.D $\quad OA \rightarrow PA$

Σχετικά με το τελεστή της ενέργειας που πρέπει να εκτελεστεί E_x, ο όρος υποδηλώνει τη πράξη "ο πράκτορας x πρέπει να κάνει", για την οποία ισχύει ο ακόλουθος κανόνας[Porn, 1977] :

E.RE $\quad (A \leftrightarrow B) / (ExA \leftrightarrow ExB)$

Από το κανόνα E.RE παράγεται το αξίωμα:

E.T $\quad E_xA \rightarrow A$

Το αξίωμα E.T υποδηλώνει ότι η ενέργεια εκτελέστηκε με επιτυχία.

2.2.1 Παραδείγματα πάνω στη θεωρία των Kanger-Lindahl

Παράδειγμα 1: Έστω α το όνομα ενός πράκτορα που έχει ως στόχο να καθαρίσει ένα δωμάτιο. Έστω B η κατάσταση που αντιπροσωπεύει ότι το δωμάτιο είναι καθαρό. Ο πράκτορας έχει την υποχρέωση να καθαρίσει το δωμάτιο σε μισή ώρα. Η έκφραση που περιγράφει την υποχρέωση που έχει ο πράκτορας είναι η εξής:

OE_aB

Η έκφραση αυτή δεν είναι η μοναδική που περιγράφει την υποχρέωση του πράκτορα και σε πολλές περιπτώσεις, ανάλογα με την υποχρέωση, μπορεί να μην περιγράφει ακριβώς την υποχρέωση του πράκτορα. Για παράδειγμα, η έκφραση αυτή μπορεί να σημαίνει είτε ότι ο πράκτορας πρέπει να φροντίσει να καθαριστεί το δωμάτιο είτε ότι κάποιος πρέπει να φροντίσει ο πράκτορας να καθαρίσει το δωμάτιο.

Μελετώντας το συγκεκριμένο παράδειγμα ακόμη περισσότερο, μπορεί κανείς να αναρωτηθεί τι γίνεται σε περίπτωση που το σύστημα είναι πολυπρακτορικό. Αν υπήρχε ένας δεύτερος πράκτορας c για τον οποίο δεν υπάρχουν πληροφορίες, χρησιμοποιώντας την modal λογική(modal logic), 3 είναι οι πιθανές περιπτώσεις:

- Ο πράκτορας c έχει την υποχρέωση να καθαρίσει το δωμάτιο
- Ο πράκτορας c επιτρέπεται αλλά δεν έχει την υποχρέωση να καθαρίσει το δωμάτιο
- Ο πράκτορας c δεν επιτρέπεται να καθαρίσει το δωμάτιο.

Χωρίς τη χρήση της modal λογικής όμως οι πιθανές περιπτώσεις μπορεί να είναι περισσότερες. Αυτός είναι και ο λόγος που είναι αναγκαία η κατασκευή μιας πλατφόρμας έτσι ώστε να είναι δυνατή η εξερεύνηση όλων των πιθανών περιπτώσεων.

Παράδειγμα 2: Έστω 2 πράκτορες a, b οι οποίοι έχουν ως στόχο να καθαρίσουν ένα δωμάτιο. Έστω C η κατάσταση που αντιπροσωπεύει ότι το δωμάτιο είναι καθαρό. Ο πράκτορας a έχει δικαίωμα να φροντίσει να καθαριστεί το δωμάτιο. Άρα, χρησιμοποιώντας την modal λογική όπως και στο παράδειγμα 1, τα δεδομένα για το πρόβλημα είναι τα εξής:

- Ο πράκτορας a επιτρέπεται να φροντίσει να καθαριστεί το δωμάτιο PE_aC.

- Ο πράκτορας b δεν επιτρέπεται να εμποδίζει τον πράκτορα a από το να φροντίσει να καθαριστεί το δωμάτιο $\neg PE_b\neg C$. Ένας άλλον ενδιαφέρον τρόπος για να γραφτεί αυτό είναι, ο πράκτορας b να απαγορεύεται να εμποδίζει τον πράκτορα a από το να φροντίσει να καθαριστεί το δωμάτιο $\neg PE_b\neg E_aC$.
- Ο πράκτορας a δεν επιτρέπεται να φροντίσει να καθαριστεί το δωμάτιο $\neg PE_aC$.
- Το δικαίωμα του πράκτορα a να φροντίσει να καθαριστεί το δωμάτιο δεν εξαρτάται από τις ενέργειες του πράκτορα b $P(E_aC \wedge \neg E_bC)$.

Άρα η έκφραση που δημιουργείται είναι η εξής:

$PE_aC \wedge \neg PE_b\neg C \wedge \neg PE_aC \wedge P(E_aC \wedge \neg E_bC)$

Η παραπάνω έκφραση είναι μια καλή προσέγγιση του τι σημαίνει η λέξη δικαίωμα στην εκφώνηση. Ωστόσο παραμένει μια προσέγγιση γιατί δεν είναι η μοναδική έκφραση που μπορεί να δημιουργηθεί, όπως φαίνεται και από τη δεύτερη περίπτωση και γιατί δεν εκφράζει τι συμβαίνει όταν ο πράκτορας b προσπαθεί αποτυχημένα να παρέμβει στις ενέργειες του πράκτορα α και πως επηρεάζονται τα δικαιώματα των πρακτόρων από αυτό[Makinson, 1986] .

Το παράδειγμα γίνεται ακόμα πιο ενδιαφέρον αν αρχίσει κάποιος και ρωτάει τι συμβαίνει με τα δικαιώματα του πράκτορα b τα οποία δεν αναφέρονται στην εκφώνηση. Κάτι τέτοιο μπορεί να μεταβάλλει και να μεγαλώσει τη παραπάνω έκφραση σε επικίνδυνο βαθμό για ένα ανθρώπινο χέρι, κάτι που δείχνει πόσο σημαντική είναι η ύπαρξη μιας πλατφόρμας που θα αυτοματοποιεί τη διαδικασία και θα απαντάει έτσι μεθοδικά και αυτόματα σε όλες τις ερωτήσεις που γεννιούνται.

Παράδειγμα 3: Ο R. Lee παρουσίασε μια παρόμοια θεωρία η οποία επιτρέπει τη περιγραφή σχέσεων όπως το αν κάτι επιτρέπεται, αν είναι υποχρεωτικό ή αν απαγορεύεται να γίνει[Lee, 1988] . Για να τη περιγράψει χρησιμοποίησε ως παράδειγμα το parking ενός πανεπιστημίου και τους κανόνες που το διέπουν. Ως πράκτορες, έθεσε τους διαχειριστές, οι οποίοι έχουν το δικαίωμα να παρκάρουν, το προσωπικό του πανεπιστημίου, το οποίο πρέπει να αποκτήσει άδεια για να παρκάρει και τους μαθητές, οι οποίοι δεν επιτρέπεται να παρκάρουν μέσα στο parking του πανεπιστημίου. Τους κανόνες αυτούς, ο R. Lee τους περιέγραψε χρησιμοποιώντας συνθήκες του τύπου if / then, οι οποίες αποτελούνται από μεταβλητές του τύπου "είναι διαχειριστής/προσωπικό/μαθητής" και "έχει/δεν έχει άδεια να παρκάρει" και ως ενέργεια έθεσε την (here, 'park') η οποία είτε μπορεί να επιτρέπεται, είτε μπορεί να είναι υποχρεωτική, είτε να απαγορεύεται.

Το πρώτο ερώτημα που γεννιέται σε αυτή τη περίπτωση είναι, αν οι 3 τιμές που μπορεί να πάρει η ενέργεια είναι αρκετές για να καλύψουν όλες τις πιθανές περιπτώσεις. Για παράδειγμα, όταν μια ενέργεια επιτρέπεται δεν είναι γνωστό αν ταυτόχρονα είναι υποχρεωτική ή αν απαγορεύεται. Επίσης, δεν είναι γνωστό τι γίνεται σε περίπτωση που υπάρχουν περισσότεροι χρήστες-πράκτορες όπως για παράδειγμα περαστικοί, αυτός που επιβλέπει το parking ή άλλοι χρήστες του parking.

Οι ερωτήσεις αυτές αν γινόντουσαν με βάση τη θεωρία των Kanger-Lindahl, θα ήταν της μορφής:

If conditions then normative position

Έτσι, μπορεί κανείς να διαπιστώσει ότι η θεωρία του R.Lee αποτελεί μια πιο απλή μορφή της θεωρίας των Kanger-Lindahl.

Το βασικό συμπέρασμα και από τα 3 παραδείγματα είναι ότι αντιπροσωπεύουν ένα ευρύ σύνολο από παρόμοια προβλήματα. Η πολυπλοκότητα του προβλήματος εξαρτάται από τον αριθμό των πρακτόρων και από τη δυνατότητα ή αδυναμία του συστήματος να απαντήσει άμεσα σε κάθε ερώτηση που γεννιέται. Για να είναι ένα πολυπρακτορικό σύστημα σε θέση να καταφέρει κάτι κάθε στιγμή, πρέπει η διαδικασία αυτή να αυτοματοποιηθεί.

2.2.2 Η θεωρία των Kanger-Lindahl

Ο βασικός σκοπός της θεωρίας των Kanger-Lindahl ήταν να βρεθεί ένας τρόπος για να είναι εφικτή η περιγραφή όλων των Κανονιστικών Θέσεων που μπορούν να αναπτυχθούν ανάμεσα σε 2 πράκτορες a,b. Αρχικά, ο S. Kanger πρότεινε την παρακάτω έκφραση, η οποία περιγράφει όλες τις σχέσεις που ανήκουν στη κατηγορία που ονόμασε "simple types of right relations":

$$\pm 0 \pm \binom{\mathrm{Ea}}{\mathrm{Eb}} \pm \mathrm{F}$$

Όπου ± αντιπροσωπεύει την επιλογή ανάμεσα σε επιβεβαίωση και άρνηση και $\binom{\mathrm{Ea}}{\mathrm{Eb}}$ την επιλογή ανάμεσα σε E_a και E_b. Άρα, προκύπτουν 16 διαφορετικές εκφράσεις. Από αυτές τις εκφράσεις ο S. Kanger παρατήρησε ότι παράγονται άλλες πιο ενδιαφέρων εκφράσεις, τις οποίες ονόμασε "atomic types of rights relation". Για να περιγράψει αυτές τις εκφράσεις, ο D. Makinson πρότεινε το παρακάτω set εκφράσεων[Makinson, 1986] :

$$\left[\!\left[\pm 0 \pm \binom{\mathrm{Ea}}{\mathrm{Eb}} \pm \mathrm{F}\right]\!\right]$$

Όπου οι αγκύλες αντιπροσωπεύουν μέγιστη σύζευξη: αν Φ είναι ένα set από εκφράσεις, τότε $[\![\Phi]\!]$ είναι η μέγιστη σύζευξη όλων των εκφράσεων που ανήκουν στο Φ και είναι λογικά συνεπείς, όπου η λέξη συνεπής αναφέρεται στη λογική των O και E_x που εισήγαγαν οι Kanger-Lindahl και η λέξη σύζευξη αναφέρεται σε ένα set από εκφράσεις, από το οποίο έχουν αφαιρεθεί όλες οι επαναλήψεις και όλες οι απλοποιήσεις.

Έτσι, οι εκφράσεις "atomic types of rights relation" του S.Kanger μπορούν να γραφτούν ως συζεύξεις 2 απλοποιημένων εκφράσεων:

$$[\![\pm\, \mathrm{O} \pm \binom{\mathrm{Ea}}{\mathrm{Eb}} \pm \mathrm{F}]\!] = [\![\pm\, \mathrm{O} \pm \mathrm{Ea} \pm \mathrm{F}]\!] * [\![\pm\, \mathrm{O} \pm \mathrm{Eb} \pm \mathrm{F}]\!]$$

Όπου η σύζευξη $[\![\pm\, \mathrm{O} \pm \mathrm{Ea} \pm \mathrm{F}]\!]$ αντιπροσωπεύει, με βάση τον S.Kanger, όλες τις Κανονιστικές Θέσεις ενός πράκτορα[Jones and Sergot, 1993]. Η σύζευξη αυτή αποτελείται από 6 εκφράσεις[Lindahl, 1977] :

(K1) PEaF ∧ PEa¬F

(K2) O¬EaF ∧ O¬Ea¬F

(K3) OEaF

(K4) PEaF ∧ P¬EaF ∧ O¬Ea¬F

(K5) OEa¬F

(K6) O¬EaF ∧ PEa¬F ∧ P¬Ea¬F

Αυτές οι 6 εκφράσεις είναι λογικά συνεπείς, αλληλοαποκλείονται, και η διάζευξή τους είναι ταυτολογία. Σε κάθε κατάσταση, μόνο μία από αυτές τις εκφράσεις μπορεί να είναι αληθής. Αργότερα, ο L. Lindahl εισήγαγε την εξής συντομογραφία:

$\mathrm{Pass}_a\mathrm{F} = \neg\mathrm{EaF} \wedge \neg\mathrm{Ea}\neg\mathrm{F}$

Όπου $Pass_aF$ είναι η κατάσταση στην οποία ο πράκτορας a δεν εκτελεί καμία ενέργεια. Άρα, η έκφραση K2 μπορεί να γραφτεί ως εξής:

(K2') $O(\neg EaF \wedge \neg Ea\neg F) = OPass_aF$

Και υποδηλώνει ότι σχετικά με την ενέργεια F, ο πράκτορας a είναι υποχρεωμένος να μη κάνει τίποτα.

Άρα, ανάμεσα σε 2 πράκτορες a,b οι συνολικές εκφράσεις που παράγονται από τις 2 συζεύξεις του S. Kanger είναι: 6*6 = 36 εκφράσεις από τις οποίες 10 δεν είναι λογικά συνεπείς, δηλαδή 26 εκφράσεις. Οι εκφράσεις αυτές είναι, επίσης, λογικά συνεπής, αλληλοαποκλείονται, η διάζευξή τους είναι ταυτολογία και σε κάθε κατάσταση μόνο μία από αυτές μπορεί να είναι αληθής. Έτσι, ο S. Kanger κατάφερε να δώσει μια πλήρης περιγραφή όλων των Κανονιστικών Θέσεων που μπορούν να αναπτυχθούν ανάμεσα σε 2 πράκτορες με βάση μια ενέργεια F.

Αργότερα, ο L. Lindahl χρησιμοποιώντας ως βάση τη θεωρία του S. Kanger παρουσίασε τον δικό του τρόπο περιγραφής των Κανονιστικών Θέσεων που μπορούν να αναπτυχθούν ανάμεσα σε 2 πράκτορες με βάση μια ενέργεια F. Για να το καταφέρει αυτό, χρησιμοποίησε διαφορετική μέγιστη σύζευξη από τον S. Kanger για να περιγράφει όλες τις Κανονιστικές Θέσεις ενός πράκτορα[Makinson, 1986] :

$$[\![\pm P \pm [\![\pm Ea \pm F]\!]]\!]$$

Η παραπάνω μέγιστη σύζευξη, σε αντίθεση με την αντίστοιχη μέγιστη σύζευξη του S. Kanger, χρησιμοποιεί τον τελεστή P και αποτελεί ένα set από μέγιστες συζεύξεις εκφράσεων του τύπου ± PA όπου A είναι μία μέγιστη σύζευξη από 3 εκφράσεις του τύπου ± Ea ± F . Αυτές οι 3 εκφράσεις αντιπροσωπεύουν τις ενέργειες που μπορεί να εκτελέσει ένας πράκτορας και είναι οι εξής[Makinson, 1986] :

(A1) EaF

(A2) $Ea\neg F$

(A3) $\neg EaF \land \neg Ea\neg F$

Όπου η 3 έκφραση αντιπροσωπεύει τη κατάσταση στην οποία ο πράκτορας δεν εκτελεί καμία ενέργεια και με βάση τη συντομογραφία του L. Lindahl μπορεί να γραφτεί και ως $Pass_aF$.

Η μέγιστη σύζευξη του L. Lindahl που περιγράφει όλες τις Κανονιστικές θέσεις ενός πράκτορα αποτελείται από 7 εκφράσεις και είναι οι εξής[Lindahl, 1977]

(T1) $PEaF \land PEa\neg F \land PPass_aF$

(T2) $PEaF \land O\neg Ea\neg F \land PPass_aF$

(T3) $PEaF \land PEa\neg F \land \neg PPass_aF$

(T4) $O\neg EaF \land PEa\neg F \land PPass_aF$

(T5) $OEaF$

(T6) $OPass_aF$

(T7) $OEa\neg F$

Στο σχήμα 2 μπορεί κανείς να παρατηρήσει ότι οι 5 από τις 6 εκφράσεις του S. Kanger είναι λογικά ισοδύναμες με 5 από τις 7 εκφράσεις του L. Lindahl.

K_1	is logically equivalent to	$(T_1 \vee T_3)$
K_2	... ...	T_6
K_3	... ...	T_5
K_4	... ...	T_2
K_5	... ...	T_7
K_6	... ...	T_4

Σχήμα 2: Normative one-agent act positions, M. Sergot, 2001, page 14, Table 1

Άρα, ανάμεσα σε 2 πράκτορες a,b οι συνολικές εκφράσεις που παράγονται από το συνδυασμό της μέγιστης σύζευξης του L. Lindahl για ένα πράκτορα είναι: 7*7 = 49 εκφράσεις, από τις οποίες 35 είναι λογικά συνεπείς και παράγονται από το ακόλουθο set εκφράσεων:

$$[\![\pm P \pm [\![\pm Ea \pm F]\!]]\!] * [\![\pm P \pm [\![\pm Eb \pm F]\!]]\!]^{\cdot\cdot}$$

Μπορεί έτσι κανείς εύκολα να επισημάνει πως η περιγραφή του L. Lindahl είναι μια βελτιωμένη έκδοση της περιγραφής του S. Kanger. Χρησιμοποιώντας ως βάση αυτό το συμπέρασμα, οι Jones-Sergot συνδύασαν τη περιγραφή των Kanger-Lindahl και δημιούργησαν το δικό τους set εκφράσεων το οποίο αποτελεί μια βελτιωμένη έκδοση της περιγραφής των Kanger-Lindahl. Συγκεκριμένα, χρησιμοποιώντας τη δυαδικότητα του τελεστή P(P = ¬O), μετέτρεψαν τη μέγιστη σύζευξη του S. Kanger για ένα πράκτορα σε:

$$[\![\pm P \pm Ea \pm F]\!]$$

Και το συνδύασαν με τις 3 καταστάσεις ενέργειας του L. Lindahl:

$[\![\pm P \pm [\![\pm Ea \pm F]\!]]\!] = [\![\pm O \pm [\![\pm Ea \pm F]\!]]\!]$

Το παραπάνω set εκφράσεων είναι αυτό που χρησιμοποίησαν οι Jones-Sergot για τη περιγραφή όλων των Κανονιστικών Θέσεων ενός πράκτορα[Jones and Sergot, 1992] [Jones and Sergot, 1993] .Το χαρακτηριστικό με τη περιγραφή των Jones-Sergot είναι ότι δίνει λύση σε πολλά προβλήματα τα οποία δεν κατάφεραν ή δεν επιχείρησαν να λύσουν οι Kanger-Lindahl. Συγκεκριμένα[Sergot, 2001] :

- Επιτρέπει τη περιγραφή των normative θέσεων σε n πράκτορες σε αντίθεση με τη περιγραφή των Kanger-Lindahl που επιτρέπει μέχρι 2 πράκτορες.
- Καλύπτει ένα ευρύτερο φάσμα από θέσεις στις οποίες μπορούν να βρεθούν οι πράκτορες και οι οποίες δεν ήταν δυνατό να περιγραφούν από τα set εκφράσεων των Kanger-Lindahl. Για παράδειγμα, έστω 2 γείτονες a, b, F η κατάσταση στην οποία ο κάθε γείτονας παρκάρει το αυτοκίνητο του μπροστά στο σπίτι του και G η κατάσταση στην οποία ο κάθε γείτονας παρκάρει στη θέση parking στην αυλή του σπιτιού του. Οι 2 γείτονες επιτρέπεται να παρκάρουν μπροστά στο σπίτι τους δηλαδή PEaF ∧ PEbF και επιτρέπεται να παρκάρουν στην αυλή τους δηλαδή PEaG ∧ PEbG. Στη θεωρία των Kanger-Lindahl οι παρακάτω Κανονιστικές Θέσεις δεν αποτελούσαν ξεχωριστή περίπτωση: P(EaF ∧ EaG) ∧ P(Eb F ∧ Eb G) και ¬P(EaF ∧ EaG) ∧ ¬P(Eb F ∧ Eb G)
- Όπως η περιγραφή του L. Lindahl αποτελεί βελτιωμένη έκδοση της περιγραφής του S. Kanger, έτσι και η περιγραφή των Jones-Sergot αποτελεί βελτιωμένη έκδοση της περιγραφής του L. Lindahl. Συγκεκριμένα, ο L. Lindahl χρησιμοποίησε 3 καταστάσεις ενέργειας για να αναπτύξει τη περιγραφή του: $[\![\pm Ea \pm F]\!]$ ενώ οι Jones-Sergot χρησιμοποίησαν 4 καταστάσεις ενέργειας:

$[\![\pm \text{Ea} \pm \text{F}]\!] * [\![\pm \text{F}]\!]$. Τις ονόμασαν "cumulative fact/act positions" και είναι η εξής:

(A1) EaF

(A2) Ea¬F

(A3a) F ∧ ¬EaF ή Pass_aF ∧ F

(A3b) ¬F ∧ ¬Ea¬F ή Pass_aF ∧ ¬F

Οι καταστάσεις A3a και A3b δεν αποτελούν διαφορετικές καταστάσεις στη περιγραφή του L. Lindahl.

Το σχήμα 3 περιγράφει τις 15 Κανονιστικές Θέσεις ενός πράκτορα a με βάση τη περιγραφή των Jones-Sergot σε αντιστοίχηση με τις αντίστοιχες 7 από την περιγραφή του L. Lindahl. Οι θέσεις αυτές παράγονται από το ακόλουθο set εκφράσεων:

$$[\![\pm \text{O} \pm [\![\pm \text{Ea} \pm \text{F}]\!] * [\![\pm \text{F}]\!]]\!]$$

$$
\begin{array}{ll}
\mathrm{T}_1 & \left\{\begin{array}{l}
\mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F) \\
\mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \neg \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F) \\
\mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \neg \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F)
\end{array}\right. \\
\mathrm{T}_2 & \left\{\begin{array}{l}
\mathrm{PE}_a F \wedge \neg \mathrm{PE}_a \neg F \wedge \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F) \\
\mathrm{PE}_a F \wedge \neg \mathrm{PE}_a \neg F \wedge \neg \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F) \\
\mathrm{PE}_a F \wedge \neg \mathrm{PE}_a \neg F \wedge \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \neg \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F)
\end{array}\right. \\
\mathrm{T}_3 & \{\mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \neg \mathrm{PPass}_a F \\
\mathrm{T}_4 & \left\{\begin{array}{l}
\neg \mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F) \\
\neg \mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \neg \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F) \\
\neg \mathrm{PE}_a F \wedge \mathrm{PE}_a \neg F \wedge \mathrm{P}(F \wedge \neg \mathrm{E}_a F) \wedge \neg \mathrm{P}(\neg F \wedge \neg \mathrm{E}_a \neg F)
\end{array}\right. \\
\mathrm{T}_5 & \{\mathrm{OE}_a F \\
\mathrm{T}_6 & \left\{\begin{array}{l}
\mathrm{OPass}_a F \wedge \mathrm{O} F \\
\mathrm{OPass}_a F \wedge \mathrm{O} \neg F \\
\mathrm{OPass}_a F \wedge \mathrm{P} F \wedge \mathrm{P} \neg F
\end{array}\right. \\
\mathrm{T}_7 & \{\mathrm{OE}_a \neg F
\end{array}
$$

Σχήμα 3: Normative one-agent cumulative fact/act positions, M. Sergot, 2001, page 16, Table 2

Αντίστοιχα, οι Κανονιστικές Θέσεις για 2 πράκτορες παράγονται από το ακόλουθο set εκφράσεων:

$$\llbracket \pm \mathrm{O} \pm \llbracket \pm \mathrm{Ea} \pm \mathrm{F} \rrbracket * \llbracket \pm \mathrm{F} \rrbracket \rrbracket * \llbracket \pm \mathrm{O} \pm \llbracket \pm \mathrm{Eb} \pm \mathrm{F} \rrbracket * \llbracket \pm \mathrm{F} \rrbracket \rrbracket$$

Με τον ίδιο τρόπο προκύπτει και το set εκφράσεων για n πράκτορες.

2.3 Η θεωρία των Jones-Sergot

Με τη θεωρία τους, οι Jones-Sergot περιέγραψαν έναν αναλυτικό τρόπο υπολογισμού όλων των Κανονιστικών Θέσεων για ένα οποιαδήποτε αριθμό πρακτόρων. Αρχικά, περιέγραψαν κάποιες έννοιες οι οποίες ήταν απαραίτητες για τη δημιουργία αυτής της θεωρίας. Ξεκίνησαν περιγράφοντας τι θεωρείται ως Διαμερίσεις ενός set από εκφράσεις δεδομένου μιας γλώσσας Λ[Sergot, 2001] .

2.3.1 Διαμερίσεις

1. Έστω P = { P1, P2,…} ένα σετ από εκφράσεις και Q μία έκφραση που ανήκει στη γλώσσα Λ. Τότε P είναι μία Λ-διαμέριση του Q αν:

- Κάθε Pi στοιχείο του P είναι λογικά συνεπές.
- Για κάθε Pi στοιχείο του P ισχύει: Q: (Pi ∈ Λ) → Q.
- Κάθε στοιχείο του P αλληλοαποκλείεται.
- Set P εξαντλεί Q.

Έτσι, προκύπτει το συμπέρασμα ότι κάθε σύζευξη της μορφής $[\![\pm \Phi]\!]$ είναι μία διαμέριση.

2. Αν P και Q είναι διαμερίσεις μιας έκφρασης R, τότε το set των συζεύξεων P*Q δεν είναι κενό και είναι διαμέριση του R.
3. Αν P και Q είναι διαμερίσεις μιας έκφρασης R, τότε P και Q είναι ίσα μόνο όταν τα στοιχεία τους σε ζεύγη είναι λογικά ίσα.
4. Αν P και Q είναι διαμερίσεις μιας έκφρασης R, τότε P ≥ Q αν ισχύει ότι: P ∈ Λ → Q
5. Αν P, Q, R είναι διαμερίσεις μιας έκφρασης S, τότε:

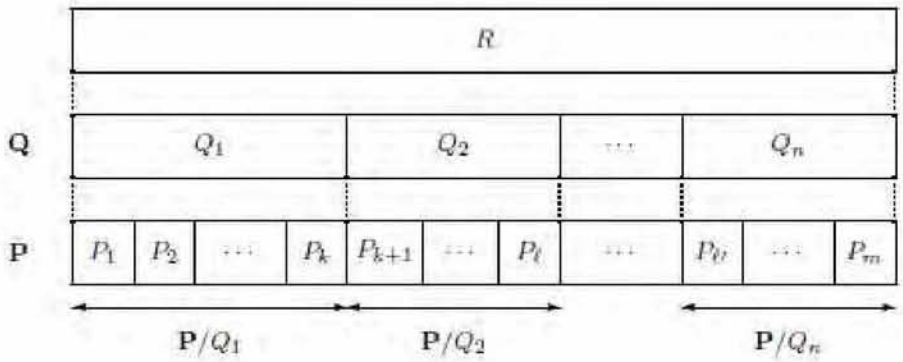
- P = Q αν P ≥ Q και Q ≥ P
- P * Q ≥ P και P * Q ≥ Q
- P * Q = P αν P ≥ Q

- Αν R ≥ P και R ≥ Q τότε R ≥ P * Q

6. Αν για set Φ1, Φ2 ισχύει Φ1 ⊆ Φ2, τότε ⟦± Φ2⟧ ≥ ⟦± Φ1⟧
7. Αν P ένα set από εκφράσεις και Q μία έκφραση τότε:

- P / Q = { Pi ∈ P | P Λ Q λογικά συνεπές }

Στο σχήμα 4 φαίνεται τι υποδηλώνει η παραπάνω σχέση

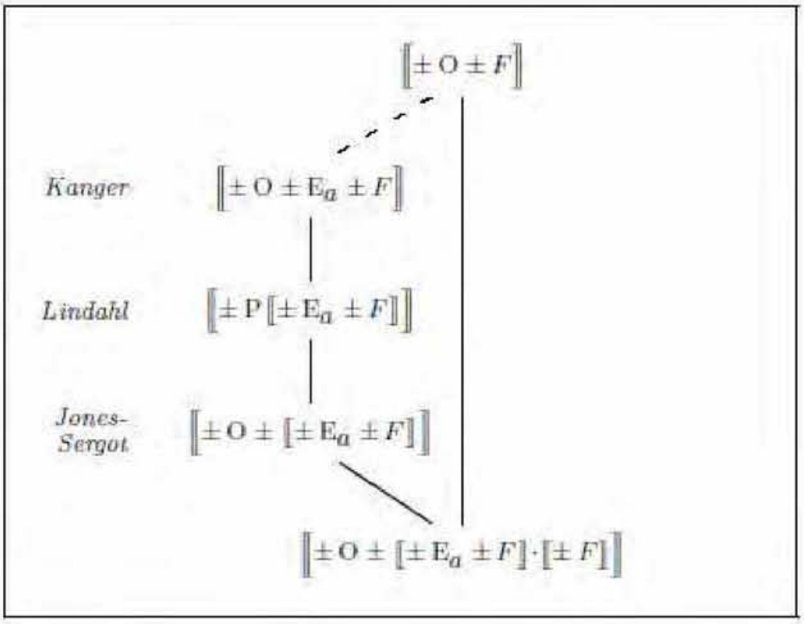


Σχήμα 4: Partitions P και Q του R με P ≥ Q, M. Sergot, 2001, page 21, Figure 2

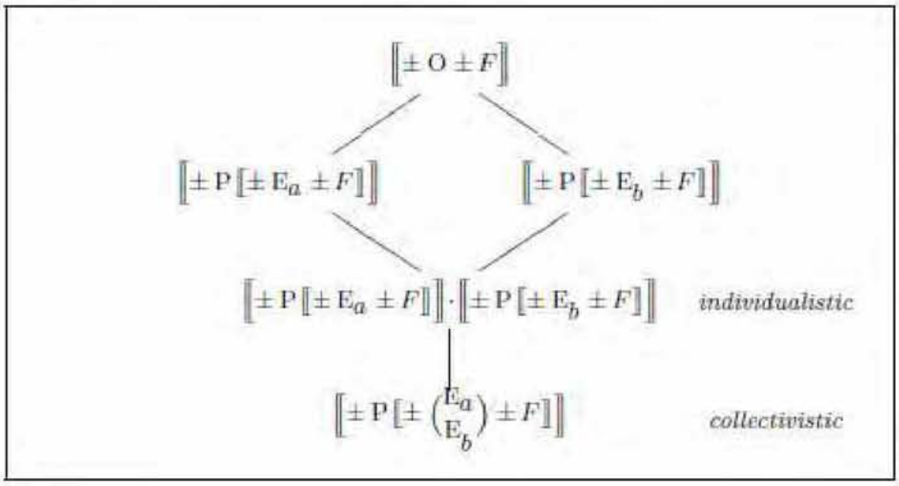
8. Έστω P και Q διαμερίσεις μιας έκφρασης R. P ≽ Q αν για κάθε Qi ∈ Q υπάρχει Pi ∈ P έτσι ώστε Pi ∈ Λ → Qi

   Το σχήμα 7 δείχνει τις σχέσεις ανάμεσα στις Κανονιστικές Θέσεις ενός πράκτορα α.

   Το σχήμα 8 δείχνει τις σχέσεις ανάμεσα στις Κανονιστικές Θέσεις του L. Lindahl για 2 πράκτορες a, b.



Σχήμα 5: Normative one-agent act positions, M. Sergot, 2001, page 21, Figure 3



Σχήμα 6: Lindahl's individualistic and collectivistic positions, M. Sergot, Normative Positions Handbook, Page 30, Figure 4

9. Έστω P, Q, R διαμερίσεις μιας έκφρασης S έτσι ώστε P $\geq$ R. Τότε για κάθε Ri $\in$ R: P * Q/ Ri = (P/Ri) * (Q/Ri).

2.3.2 Οι Κανονιστικές Θέσεις των Jones-Sergot

Χρησιμοποιώντας τη παραπάνω θεωρία για τις διαμερίσεις, οι Jones-Sergot εισήγαγαν ένα σαφή και αποτελεσματικό τρόπο υπολογισμού για μέγιστες συζεύξεις, καθώς και ένα τρόπο εντοπισμού των set των εκφράσεων που παράγουν τις μέγιστες συζεύξεις που είναι απαραίτητες για τη λύση κάποιου προβλήματος.sergot2001

Έστω ένα set από μέγιστες συζεύξεις του τύπου $[\![\pm O \pm \mathrm{A}]\!]$ όπου A είναι μία Λ-διαμέριση. Λόγω της δυαδικότητας του τελεστή O οι παραπάνω μέγιστες συζεύξεις μπορούν να γραφούν και ως $[\![\pm P \pm \mathrm{A}]\!]$. Τότε, οι παραπάνω μέγιστες συζεύξεις αποτελούνται από συζεύξεις του τύπου:

$\pm PA1 \wedge \ldots \wedge \mathrm{PAj} \wedge \ldots \wedge \pm PAn$

Για τα οποία ισχύει ότι για κάθε Ai $\in$ A, υπάρχει ένας συζευκτέος του τύπου PAi ή ¬PAi και τουλάχιστον ένας συζευκτέος του τύπου PAj. Οι συζεύξεις αυτές γράφονται σε συντομογραφία ως πA. Αντίστοιχα, με π^{+}A γράφονται σε συντομογραφία οι συζεύξεις του τύπου PAi και με π^{-}A γράφονται σε συντομογραφία οι συζεύξεις του τύπου ¬PAi. Άρα, με βάση αυτή τη θεωρία, οι μέγιστες συζεύξεις του τύπου $[\![\pm O \pm \mathrm{A}]\!]$ μπορούν να γραφτούν και ως $[\![\pm PA]\!]$.

Έτσι, αν κάποιος χρειάζεται να επισημάνει ένα από τα στοιχεία πA των $[\![\pm\, O \,\pm$ $A]\!]$, αρκεί να επισημάνει τα στοιχεία που είναι του τύπου $\pi^{+}A$.

Προκειμένου να γίνει πλήρως κατανοητό το πώς χρησιμοποιείται η παραπάνω θεωρία για τον υπολογισμό των Κανονιστικών Θέσεων, θα χρησιμοποιηθεί ως παράδειγμα η θεωρία των Jones-Parent, οι οποίοι μελέτησαν τις Κανονιστικές-Ενημερωτικές Θέσεις, οι οποίες απευθύνονται σε σχέσεις-άδειες του τύπου «άδεια στη σιωπή», «άδεια στην γνώση μιας πληροφορίας» και «άδεια στην απόκρυψη μιας πληροφορίας». Jones and Parent [2008] Έστω IjA η συντομογραφία που αντιπροσωπεύει ότι ο πράκτορας j πληροφορήθηκε ότι A. Έστω OkA η συντομογραφία που αντιπροσωπεύει ότι είναι υποχρεωτικό ο πράκτορας k να A. Οι μέγιστες συζεύξεις των Jones-Parent που παράγουν τις *Normative-Informational* θέσεις είναι οι παρακάτω:

$$[\![\pm\, \mathrm{Ok} \,\pm\, [\![\pm\, \mathrm{Ij} \,\pm\, \mathrm{A}]\!]]\!]$$

Στο set $[\![\pm\, \mathrm{Ij} \,\pm\, \mathrm{A}]\!]$ υπάρχουν 4 *Informational* θέσεις:

(I1) $\mathrm{IjA} \wedge \neg\mathrm{Ij}\neg\mathrm{A}$

(I2) $\mathrm{Ij}\neg\mathrm{A} \wedge \neg\mathrm{IjA}$

(I3) $\neg\mathrm{IjA} \wedge \neg\mathrm{Ij}\neg\mathrm{A}$

(I4) $\mathrm{IjA} \wedge \mathrm{Ij}\neg\mathrm{A}$

Άρα, το set $[\![\pm\, \mathrm{Ij} \,\pm\, \mathrm{A}]\!]$ αποτελείται από $2^4 - 1 = 15$ υποσύνολα, δηλαδή από 15 Κανονιστικές-Ενημερωτικές θέσεις.

Αντίστοιχα, έστω ένα set από μέγιστες συζεύξεις του τύπου $[\![\pm\, O \,\pm\, \mathrm{A}]\!]$ / πB με $A \geq B$, τα οποία αποτελούν και τη βάση των αυτοματοποιημένων μεθόδων του M. Sergot που θα χρησιμοποιηθούν για την ανάπτυξη της πλατφόρμας για τον υπολογισμό των Κανονιστικών Θέσεων. Έστω A,B σύνολα για τα οποία ισχύει ότι $A \geq B$ kai πB είναι στοιχείο της σύζευξης $[\![\pm 0 \pm B]\!]$ το οποίο είναι της μορφής:

$\neg P\ B1 \wedge \ldots \wedge \neg PBk \wedge PBk{+}1 \wedge \ldots \wedge PBn \quad (k \geq 1)$

Συγκεκριμένα, $\pi^{+}B$ = [Bκ+1, …, Bn] και $\pi^{-}B$ = [B1, …, Bκ]. Τότε, κάθε στοιχείο του set $[\![\pm 0 \pm A]\!]$ / πB είναι μία σύζευξη του τύπου:

$\neg P\ B1 \wedge \ldots \wedge \neg PBk \wedge \pi(A / B\kappa{+}1) \wedge \ldots \wedge \pi (A / Bn)$

Έτσι, δημιουργείται το συμπέρασμα ότι, αφού $A \geq B$ ισχύει ότι $[\![\pm 0 \pm A]\!] \geq [\![\pm 0 \pm B]\!]$

2.3.3 Παράδειγμα υπολογισμού Κανονιστικών Θέσεων

Έστω ότι υπάρχουν 2 κτίρια και η ενέργεια F υποδηλώνει ότι υπάρχει ανάμεσα τους ένα parking. Στόχος είναι να υπολογιστούν οι υποχρεώσεις ενός πράκτορα a. Για να επιτευχθεί αυτό, πρέπει να καθοριστούν οι Κανονιστικές Θέσεις του τύπου $[\![\pm O \pm [\![\pm Ea \pm F]\!] * [\![\pm F]\!]]\!]$ οι οποίες είναι συνεπής με την έκφραση OF, δηλαδή:

$[\![\pm O \pm [\![\pm Ea \pm F]\!] * [\![\pm F]\!]]\!] / OF$

Η έκφραση OF αποτελείται από τη σύζευξη $PF \wedge \neg P\ \neg F$. Άρα, το παραπάνω set θα αποτελείται από σύζευξη του τύπου $\neg P\ \neg F \wedge C$ όπου C είναι μία σύζευξη του τύπου $\pi([\![\pm Ea \pm F]\!] * [\![\pm F]\!] \ / \ F)$. Από τη θεωρία των Sergot-Jones είναι γνωστό ότι η έκφραση μέσα στη παρένθεση μπορεί να γραφτεί και ως [F $\wedge$ EaF, F $\wedge$ ¬EaF] = [EaF, F $\wedge$ ¬EaF]. Έτσι, προκύπτει το συμπέρασμα ότι το set αποτελείται από 3 υποσύνολα, άρα υπάρχουν και 3 Κανονιστικές Θέσεις οι οποίες είναι οι εξής:

¬P ¬F $\wedge$ PEaF $\wedge$ ¬P(F $\wedge$ ¬EaF) = OEaF

¬P ¬F $\wedge$ ¬PEaF $\wedge$ P(F $\wedge$ ¬EaF) = OF $\wedge$ ¬PEaF

¬P ¬F $\wedge$ PEaF $\wedge$ P(F $\wedge$ ¬EaF) = OF $\wedge$ PEaF $\wedge$ P¬EaF

Αντίστοιχα, μπορούν να υπολογιστούν και ποιες Κανονιστικές Θέσεις, για 2 πράκτορες a,b είναι συνεπείς με την έκφραση OEaF

Ο υπολογισμός των Κανονιστικών Θέσεων θα γίνει χρησιμοποιώντας το set:

$\llbracket \pm \mathrm{O} \pm \llbracket \pm \mathrm{Ea} \pm \mathrm{F} \rrbracket * \llbracket \pm \mathrm{Eb} \pm \mathrm{F} \rrbracket * \llbracket \pm \mathrm{F} \rrbracket \rrbracket$ / OEaF

Άρα, το παραπάνω set θα αποτελείται από σύζευξη του τύπου OEaF $\wedge$ C όπου C είναι μία σύζευξη του τύπου π($\llbracket \pm \mathrm{Ea} \pm \mathrm{F} \rrbracket * \llbracket \pm \mathrm{Eb} \pm \mathrm{F} \rrbracket \llbracket \pm \mathrm{F} \rrbracket$ / EaF). Από τη θεωρία των Sergot-Jones είναι γνωστό ότι η έκφραση μέσα στη παρένθεση μπορεί να γραφτεί και ως ($\llbracket \pm \mathrm{Ea} \pm \mathrm{F} \rrbracket$ / EaF) * ($\llbracket \pm \mathrm{Eb} \pm \mathrm{F} \rrbracket$ / EaF) = [EaF $\wedge$ EbF, EaF $\wedge$ ¬EbF]. Έτσι, προκύπτει το συμπέρασμα ότι το set αποτελείται από 3 υποσύνολα, άρα υπάρχουν και 3 Κανονιστικές Θέσεις οι οποίες είναι οι εξής:

OEaF $\wedge$ PEbF $\wedge$ ¬P¬EbF = OEaF $\wedge$ OEbF

OEaF $\wedge$ ¬PEbF $\wedge$ P¬EbF = OEaF $\wedge$ O¬EbF

OEaF $\wedge$ PEbF $\wedge$ P¬EbF = OEaF $\wedge$ PEbF $\wedge$ P¬EbF

Ακολουθώντας την ίδια λογική, μπορεί κανείς να υπολογίσει τις Κανονιστικές Θέσεις για έναν οποιονδήποτε αριθμό πρακτόρων. Ωστόσο, όσο αυξάνεται ο αριθμός των πρακτόρων και ο αριθμός των ενεργειών τόσο περισσότερες και πιο πολύπλοκες γίνονται οι πράξεις που απαιτούνται. Συγκεκριμένα, αν ο αριθμός των πρακτόρων είναι m και ο αριθμός των διαφορετικών ενεργειών είναι n, ο αριθμός των Κανονιστικών Θέσεων θα είναι $2^{2(m+1)n} - 1$. Σκοπός, λοιπόν, αυτής της εργασίας είναι η δημιουργία μίας πλατφόρμας που θα αυτοματοποιεί τη διαδικασία υπολογισμού των Κανονιστικών Θέσεων.

ΚΕΦΑΛΑΙΟ 3 – ΣΧΕΔΙΑΣΜΟΣ ΤΗΣ ΠΛΑΤΦΟΡΜΑΣ

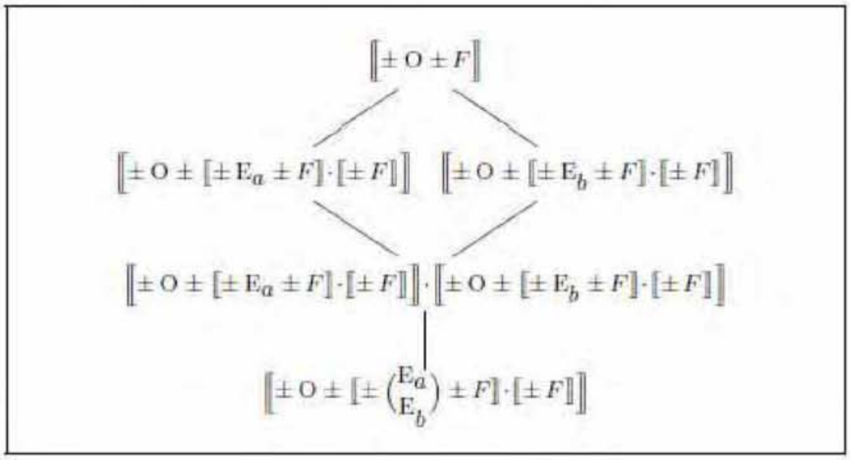
Τώρα που ολοκληρώθηκε η αναφορά της θεωρίας πάνω στην οποία βασίστηκε η ιδέα για τη παραγωγή μιας πλατφόρμας που θα αυτοματοποιεί τη διαδικασία υπολογισμού των Κανονιστικών Θέσεων, μπορεί κανείς να συνειδητοποιήσει καλύτερα την αναγκαιότητα ύπαρξης μιας τέτοιας πλατφόρμας. Συγκεκριμένα, ενώ ο υπολογισμός των Κανονιστικών Θέσεων είναι πολύ εύκολος να γίνει με το χέρι, μιας και χρησιμοποιώντας το set εκφράσεων του M. Sergot οι Κανονιστικές Θέσεις που προκύπτουν για ένα πράκτορα είναι 15, ο υπολογισμός των Κανονιστικών Θέσεων για 2 ή περισσότερους πράκτορες είναι επίσης μια εύκολη διαδικασία, αλλά περισσότερο μεγαλύτερη και επίπονη σε σχέση με ένα πράκτορα.

Για τον υπολογισμό των Κανονιστικών Θέσεων για 2 πράκτορες a, b απαιτείται ο υπολογισμός 15 * 15 = 225 εκφράσεων. Είναι εύκολο, λοιπόν, να συνειδητοποιήσει κανείς πόσο μεγάλη είναι η διαδικασία υπολογισμού με το χέρι. Αντίθετα, χρησιμοποιώντας μία έξυπνη πλατφόρμα που θα αυτοματοποιεί τη διαδικασία, χρησιμοποιώντας ως είσοδο τις 15 Κανονιστικές Θέσεις του M. Sergot, η διαδικασία αυτή μπορεί να γίνει πιο εύκολη και κυρίως πιο γρήγορη.

Για τη δημιουργία της πλατφόρμας, χρησιμοποιήθηκε η γλώσσα προγραμματισμού Java, και η ανάπτυξη του κώδικα έγινε χρησιμοποιώντας τον editor NetBeans.

3.1 Ο Ψευδοκώδικας

Όπως αναφέρθηκε, απαραίτητη προϋπόθεση για την ανάπτυξη της πλατφόρμας ήταν η χρήση των 15 Κανονιστικών Θέσεων του M. Sergot. Οι 15 αυτές εκφράσεις συνοψίζονται στο σχήμα 3. Επίσης, στο σχήμα 7 βρίσκεται το σχεδιάγραμμα που επεξηγεί επιγραμματικά τη διαδικασία υπολογισμού των Κανονιστικών Θέσεων για 2 πράκτορες, πάνω στο οποίο βασίστηκε και η ανάπτυξη του κώδικα για n πράκτορες[Paper Sergot].



Σχήμα 7: Positions για 2 πράκτορες a, b και ένα state of affairs F, M. Sergot, 2001, page 28, Figure 4

Η συγκεκριμένη πλατφόρμα, είναι εμπνευσμένη από το υπολογιστικό σύστημα Norman-G που πρότεινε ο M. Sergot για τον υπολογισμό των Κανονιστικών Θέσεων για ένα m αριθμό πρακτόρων και ένα n αριθμό από state of affairs.

Η πλατφόρμα, στην τωρινή κατάσταση της, μπορεί να υπολογίζει όλες τις Κανονιστικές Θέσεις για ένα n αριθμό πρακτόρων που καθορίζεται από το χρήστη στην αρχή του προγράμματος και ένα state of affairs F. Ο ψευδοκώδικας που απαιτείται για την ανάπτυξη αυτής της πλατφόρμας είναι ο εξής:

- Φτιάξε μια δομή που θα περιέχει τις 15 Κανονιστικές Θέσεις του M. Sergot με το σύμβολο # στη θέση του ονόματος του πράκτορα.
- Φτιάξε ένα constructor ο οποίος θα παίρνει ως είσοδο το όνομα του πράκτορα.
    - Θα παράγει αντικείμενα-πράκτορες τα οποία θα αποτελούνται από το όνομα του πράκτορα.
    - Θα περιέχει μία άδεια δομή
    - Ένας βρόχος θα διατρέχει τη δομή με τις 15 Κανονιστικές Θέσεις και θα αντικαθιστά το σύμβολο # με το όνομα του πράκτορα αποθηκεύοντας το αποτέλεσμα στην άδεια δομή.
- Φτιάξε ένα constructor ο οποίος θα παίρνει ως είσοδο 2 αντικείμενα-πράκτορες.
    - Θα παράγει αντικείμενα-πράκτορες τα οποία θα αποτελούνται από τα ονόματα των 2 πρακτόρων του κάθε αντικειμένου ενωμένα.
    - Θα περιέχει μία άδεια δομή
    - Δύο βρόχοι θα διατρέχουν τις δομές του κάθε αντικειμένου, υπολογίζοντας όλες τις δυνατές συζεύξεις μεταξύ των στοιχείων των δομών και θα αποθηκεύεται το αποτέλεσμα στην άδεια δομή.

3.2 Ανάλυση της πλατφόρμας

Αρχικά, είναι απαραίτητη η ύπαρξη μιας δομής η οποία θα περιέχει τις 15 Κανονιστικές Θέσεις του M. Sergot, με το σύμβολο # στη θέση του ονόματος του πράκτορα. Ο λόγος που γίνεται αυτό είναι ότι:

- Οι 15 Κανονιστικές Θέσεις για ένα πράκτορα a και ένα state of affairs F, είναι εύκολο να υπολογιστούν με το χέρι αλλά αρκετά δύσκολο να υπολογιστούν με κώδικα. Οι 15 Κανονιστικές Θέσεις έχουν καθοριστεί από τον M. Sergot μετά από πολλά κλαδέματα και απλοποιήσεις περισσότερων Κανονιστικών Θέσεων προκειμένου να υπολογιστούν οι λιγότερες δυνατές Κανονιστικές Θέσεις. Ωστόσο, αυτά τα κλαδέματα και οι απλοποιήσεις είναι δύσκολο να μεταφραστούν σε ψευδοκώδικα
- Οι 15 Κανονιστικές Θέσεις του M. Sergot είναι ήδη υπολογισμένες και γνωστές και η δημιουργία κώδικα για τον εκ νέου υπολογισμό τους θα λειτουργούσε κυρίως ως επιβάρυνση στο χρόνο εκτέλεσης της πλατφόρμας.

Η πρώτη συνάρτηση είναι στην ουσία ένας constructor που χρησιμοποιείται για να δημιουργούνται αντικείμενα-πράκτορες. Στη συνάρτηση αυτή, αρχικά, αποθηκεύεται σε μία μεταβλητή το όνομα του πράκτορα και στη συνέχεια, αποθηκεύονται σε μία δομή οι 15 Κανονιστικές Θέσεις του M. Sergot για ένα πράκτορα. Τέλος, γίνεται αντικατάσταση του χαρακτήρα # κατά μήκος της δομής με το χαρακτήρα που έδωσε ο χρήστης ως όνομα για το πράκτορα. Κύρια λειτουργία αυτής της συνάρτησης είναι η δημιουργία των Κανονιστικών Θέσεων για ένα πράκτορα. Με βάση τη θεωρία, η λειτουργία αυτή είναι η εξής:

$$[\![\pm O \pm [\![\pm Ea \pm F]\!] * [\![\pm F]\!]]\!]$$

Η επόμενη συνάρτηση είναι επίσης ένας constructor, ο οποίος παίρνει ως είσοδο 2 αντικείμενα-πράκτορες και πραγματοποιεί τους απαραίτητους υπολογισμούς που απαιτούνται για την δημιουργία των Κανονιστικών Θέσεων για 2 πράκτορες. Συγκεκριμένα, αποθηκεύονται σε μία μεταβλητή τα ονόματα των 2 πρακτόρων ενωμένα με το τελεστή Λ και χρησιμοποιούνται 2 βρόχοι οι οποίοι διατρέχουν τη δομή του κάθε πράκτορα με τις 15 Κανονιστικές Θέσεις και υπολογίζουν όλους τους πιθανούς συνδυασμούς ανάμεσα στις 15 Κανονιστικές Θέσεις του ενός πράκτορα και τις 15 Κανονιστικές Θέσεις του άλλου. Κύρια λειτουργία αυτής της συνάρτησης είναι η δημιουργία των Κανονιστικών Θέσεων για 2 πράκτορες. Με βάση τη θεωρία, η λειτουργία αυτή είναι η εξής:

$$[\![\pm O \pm [\![\pm Ea \pm F]\!] * [\![\pm F]\!]]\!] * [\![\pm O \pm [\![\pm Eb \pm F]\!] * [\![\pm F]\!]]\!]$$

Η επόμενη συνάρτηση έχει ως κύρια λειτουργία να μεταφέρει τις Κανονιστικές Θέσεις από τις δομές των πρακτόρων σε ένα αρχείο με όνομα που δίνεται από το χρήστη κατά την αρχή του προγράμματος.

Τέλος, η βασική συνάρτηση της πλατφόρμας είναι η συνάρτηση μέσα στην οποία καλούνται όλες οι υπόλοιπες συναρτήσεις προκειμένου να λειτουργήσει και καθορίζει τη ροή λειτουργίας της πλατφόρμας. Αρχικά, η πλατφόρμα ρωτάει τον χρήστη ποιός είναι ο αριθμός των πρακτόρων και αποθηκεύει τον αριθμό που δίνει ο χρήστης σε μια μεταβλητή. Στη συνέχεια, ρωτάει το χρήστη το όνομα του αρχείου μέσα στο οποίο θα αποθηκεύονται οι Κανονιστικές Θέσεις που υπολογίζονται από τη πλατφόρμα και αποθηκεύει το όνομα που δίνει ο χρήστης σε μία μεταβλητή. Έπειτα, δηλώνει στο χρήστη ότι αρχίζει ο υπολογισμός των Κανονιστικών Θέσεων, δημιουργεί τα ονόματα των πρακτόρων, ανάλογα με τον αριθμό που δήλωσε ο χρήστης και αποθηκεύει τα ονόματα σε μία δομή. Τέλος, υπολογίζει τις 15 Κανονιστικές Θέσεις για κάθε πράκτορα ξεχωριστά, και τις μεταφέρει στο αρχείο.

Σε αυτό το σημείο, σε περίπτωση που ο αριθμός των πρακτόρων είναι ίσος ή μεγαλύτερος του 2, το πρόγραμμα χρησιμοποιεί την δεύτερη συνάρτηση για να υπολογίσει όλες τις Κανονιστικές θέσεις για όλους τους πράκτορες συνολικά. Συγκεκριμένα, χρησιμοποιεί ένα βρόχο για να υπολογίσει πρώτα τις Κανονιστικές Θέσεις ανάμεσα στους 2 πρώτους πράκτορες, μεταφέρει το αποτέλεσμα στο αρχείο και συνεχίζει υπολογίζοντας τις Κανονιστικές Θέσεις ανάμεσα στους 2 πρώτους πράκτορες και στον επόμενο πράκτορα μέχρι να ολοκληρωθεί ο υπολογισμός για όλους τους πράκτορες. Τέλος, ενημερώνει το χρήστη πότε ολοκληρώνεται ο υπολογισμός των Κανονιστικών Θέσεων, σε ποιό αρχείο αποθηκεύτηκαν και πότε τερματίζεται το πρόγραμμα.

Τέλος, η πλατφόρμα ελέγχει αν ο χρήστης εισάγει λανθασμένα δεδομένα ή αν παρουσιαστεί πρόβλημα κατά δημιουργία του αρχείου.

ΚΕΦΑΛΑΙΟ 4 – ΕΚΤΕΛΕΣΗ ΤΗΣ ΠΛΑΤΦΟΡΜΑΣ

Στο κεφάλαιο αυτό θα δοθούν αναλυτικά παραδείγματα και στιγμιότυπα από την εκτέλεση του προγράμματος. Επιγραμματικά, η εκτέλεση του προγράμματος ακολουθεί το εξής μοτίβο:

- Το πρόγραμμα ζητάει από το χρήστη τον αριθμό των πρακτόρων.
- Το πρόγραμμα ζητάει από το χρήστη το όνομα του αρχείου στο οποίο θα αποθηκευτούν οι Κανονιστικές Θέσεις όταν υπολογιστούν.
- Το πρόγραμμα ενημερώνει το χρήστη ότι ξεκίνησε τον υπολογισμό των Κανονιστικών Θέσεων για κάθε πράκτορα.
- Το πρόγραμμα ενημερώνει το χρήστη ότι ξεκίνησε τον υπολογισμό των Κανονιστικών Θέσεων για όλους τους πράκτορες.
- Το πρόγραμμα ενημερώνει το χρήστη ότι ολοκληρώθηκε ο υπολογισμός των Κανονιστικών Θέσεων και ότι έγινε με επιτυχία η μεταφορά τους στο αρχείο.
- Το πρόγραμμα ενημερώνει το χρήστη ότι ολοκληρώθηκε η λειτουργία του.

4.1 Παραδείγματα εκτέλεσης της πλατφόρμας

4.1.1 Παράδειγμα εκτέλεσης για ένα πράκτορα

Το πρώτο παράδειγμα αφορά τον υπολογισμό των Κανονιστικών Θέσεων ενός πράκτορα a για ένα state of affairs F. Αρχικά, όταν ο χρήστης εκτελέσει τη πλατφόρμα, εμφανίζεται στην κονσόλα το ακόλουθο μήνυμα:

" How many agents ? "

Εδώ, ο χρήστης καλείται να δώσει τον αριθμό των πρακτόρων, που στο συγκεκριμένο παράδειγμα είναι ένας. Άρα, επιλέγει στο πληκτρολόγιο 1 και πατάει enter:

" How many agents ? 1 "

Στη συνέχεια, εμφανίζεται το ακόλουθο μήνυμα:

" Output filename ? "

Εδώ, ο χρήστης καλείται να δώσει το όνομα του αρχείου μέσα στο οποίο θα αποθηκευτούν οι Κανονιστικές Θέσεις όταν υπολογιστούν. Στο συγκεκριμένο παράδειγμα το όνομα είναι " norm1 ":

" Output filename ? norm1"

Τέλος, η πλατφόρμα εμφανίζει τα ακόλουθα μηνύματα όταν ολοκληρώσει τη διαδικασία που αντιπροσωπεύει το καθένα:

- "Creating normative positions for each agent " όταν ξεκινήσει τον υπολογισμό των Κανονιστικών Θέσεων για κάθε πράκτορα ξεχωριστά.
- "Creating normative positions for all agents " όταν ολοκληρώσει το παραπάνω υπολογισμό και ξεκινήσει τον υπολογισμό των Κανονιστικών Θέσεων για όλους τους πράκτορες ταυτόχρονα.
- " All normative positions have been created and saved to file norm.txt " όταν ολοκληρώσει τον υπολογισμό όλων των Κανονιστικών Θέσεων και δημιουργήσει επιτυχώς το κατάλληλο .txt αρχείο.
- "Program ended. " όταν τερματιστεί η πλατφόρμα.

Σε περίπτωση λανθασμένης εισόδου κατά τη δήλωση του αριθμού των πρακτόρων η πλατφόρμα εμφανίζει το ακόλουθο μήνυμα και τερματίζεται:

"Illegal number, try again"

Σε περίπτωση που παρουσιαστεί πρόβλημα κατά τη δημιουργία του αρχείου, η πλατφόρμα ενημερώνει το χρήστη.

Η έξοδος της πλατφόρμας για το συγκεκριμένο παράδειγμα είναι ένα .txt αρχείο που ονομάζεται norm και περιέχει τις ακόλουθες Κανονιστικές Θέσεις:

Normative Positions for agent(s): a

1. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F)

2. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F)

3. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F)

4. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F)

5. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F)

6. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F)

7. P Ea F Λ P Ea ~F Λ ~P Passa F

8. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F)

9. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F)

10. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F)

11. Ea F

12. O Passa F Λ O F

13. O Passa F Λ O ~F

14. O Passa F Λ P F Λ P ~F

15. O Ea ~F

4.1.2 Παράδειγμα εκτέλεσης για 2 πράκτορες

Το δεύτερο παράδειγμα αφορά τον υπολογισμό των Κανονιστικών Θέσεων 2 πρακτόρων a, b για ένα state of affairs F. Η εκτέλεση της πλατφόρμας ακολουθεί το ίδιο μοτίβο με τη μόνη διαφορά ότι η είσοδος του χρήστη στον αριθμό των πρακτόρων είναι " 2 ". Η έξοδος του προγράμματος για το συγκεκριμένο πρόγραμμα είναι ένα .txt αρχείο που ονομάζεται norm και περιέχει τις ακόλουθες Κανονιστικές Θέσεις:

Normative Positions for agent(s): a

1. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F)

2. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F)

3. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F)

4. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F)

5. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F)

6. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F)

7. P Ea F Λ P Ea ~F Λ ~P Passa F

8. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F)

9. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F)

10. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F)

11. Ea F

12. O Passa F Λ O F

13. O Passa F Λ O ~F

14. O Passa F Λ P F Λ P ~F

15. O Ea ~F

Normative Positions for agent(s): b

1. P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

2. P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

3. P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

4. P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

5. P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

6. P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

7. P Eb F Λ P Eb ~F Λ ~P Passb F

8. ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

9. ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

10. ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

11. Eb F

12. O Passb F Λ O F

13. O Passb F Λ O ~F

14. O Passb F Λ P F Λ P ~F

15. O Eb ~F

Normative Positions for agent(s): aΛb

1. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

2. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

3. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

4. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

5. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

6. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

7. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

8. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

9. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

10. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

11. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ Eb F

12. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O F

13. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O ~F

14. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ P F Λ P ~F

15. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Eb ~F

16. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

17. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

18. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

19. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

20. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

21. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

22. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

23. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

24. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

25. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

26. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ Eb F

27. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ O F

28. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ O ~F

29. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ P F Λ P ~F

30. P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Eb ~F

31. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

32. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

33. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

34. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

35. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

36. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

37. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

38. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

39. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

40. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

41. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ Eb F

42. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ O Passb F Λ O F

43. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ O Passb F Λ O ~F

44. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ O Passb F Λ P F Λ P ~F

45. P Ea F Λ P Ea ~F Λ P(F Λ ~Ea F) Λ ~ P(~F Λ ~Ea ~F) Λ O Eb ~F

46. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

47. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

48. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

49. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

50. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

51. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

52. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

53. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

54. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

55. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

56. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ Eb F

57. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ O F

58. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ O ~F

59. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ P F Λ P ~F

60. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Eb ~F

61. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

62. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

63. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

64. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

65. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

66. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

67. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

68. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

69. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

70. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

71. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ Eb F

72. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O F

73. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O ~F

74. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ P F Λ P ~F

75. P Ea F Λ ~ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Eb ~F

76. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

77. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

78. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

79. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

80. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

81. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

82. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

83. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

84. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

85. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

86. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ Eb F

87. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O F

88. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O ~F

89. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Passb F Λ P F Λ P ~F

90. P Ea F Λ ~ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Eb ~F

91. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

92. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

93. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

94. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

95. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

96. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

97. P Ea F Λ P Ea ~F Λ ~P Passa F Λ P Eb F Λ P Eb ~F Λ ~P Passb F

98. P Ea F Λ P Ea ~F Λ ~P Passa F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

99. P Ea F Λ P Ea ~F Λ ~P Passa F Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

100. P Ea F Λ P Ea ~F Λ ~P Passa F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

101. P Ea F Λ P Ea ~F Λ ~P Passa F Λ Eb F

102. P Ea F Λ P Ea ~F Λ ~P Passa F Λ O Passb F Λ O F

103. P Ea F Λ P Ea ~F Λ ~P Passa F Λ O Passb F Λ O ~F

104. P Ea F Λ P Ea ~F Λ ~P Passa F Λ O Passb F Λ P F Λ P ~F

105. P Ea F Λ P Ea ~F Λ ~P Passa F Λ O Eb ~F

106. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

107. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

108. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

109. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

110. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

111. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

112. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

113. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

114. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

115. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

116. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ Eb F

117. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ O F

118. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ O ~F

119. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Passb F Λ P F Λ P ~F

120. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ P(~F Λ ~Ea ~F) Λ O Eb ~F

121. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

122. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

123. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

124. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

125. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

126. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

127. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

128. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

129. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

130. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

131. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ Eb F

132. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O F

133. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O ~F

134. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Passb F Λ P F Λ P ~F

135. ~P Ea F Λ P Ea ~F Λ ~ P(F Λ ~ Ea F) Λ P(~F Λ ~ Ea ~F) Λ O Eb ~F

136. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

137. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

138. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

139. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

140. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

141. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

142. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ P Eb F Λ P Eb ~F Λ ~P Passb F

143. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

144. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

145. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

146. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ Eb F

147. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O F

148. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Passb F Λ O ~F

149. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Passb F Λ P F Λ P ~F

150. ~P Ea F Λ P Ea ~F Λ P(F Λ ~ Ea F) Λ ~ P(~F Λ ~ Ea ~F) Λ O Eb ~F

151. Ea F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

152. Ea F Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

153. Ea F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

154. Ea F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

155. Ea F Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

156. Ea F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

157. Ea F Λ P Eb F Λ P Eb ~F Λ ~P Passb F

158. Ea F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

159. Ea F Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

160. Ea F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

161. Ea F Λ Eb F

162. Ea F Λ O Passb F Λ O F

163. Ea F Λ O Passb F Λ O ~F

164. Ea F Λ O Passb F Λ P F Λ P ~F

165. Ea F Λ O Eb ~F

166. O Passa F Λ O F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

167. O Passa F Λ O F Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

168. O Passa F Λ O F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

169. O Passa F Λ O F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

170. O Passa F Λ O F Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

171. O Passa F Λ O F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

172. O Passa F Λ O F Λ P Eb F Λ P Eb ~F Λ ~P Passb F

173. O Passa F Λ O F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

174. O Passa F Λ O F Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

175. O Passa F Λ O F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

176. O Passa F Λ O F Λ Eb F

177. O Passa F Λ O F Λ O Passb F Λ O F

178. O Passa F Λ O F Λ O Passb F Λ O ~F

179. O Passa F Λ O F Λ O Passb F Λ P F Λ P ~F

180. O Passa F Λ O F Λ O Eb ~F

181. O Passa F Λ O ~F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

182. O Passa F Λ O ~F Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

183. O Passa F Λ O ~F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

184. O Passa F Λ O ~F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

185. O Passa F Λ O ~F Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

186. O Passa F Λ O ~F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

187. O Passa F Λ O ~F Λ P Eb F Λ P Eb ~F Λ ~P Passb F

188. O Passa F Λ O ~F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

189. O Passa F Λ O ~F Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

190. O Passa F Λ O ~F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

191. O Passa F Λ O ~F Λ Eb F

192. O Passa F Λ O ~F Λ O Passb F Λ O F

193. O Passa F Λ O ~F Λ O Passb F Λ O ~F

194. O Passa F Λ O ~F Λ O Passb F Λ P F Λ P ~F

195. O Passa F Λ O ~F Λ O Eb ~F

196. O Passa F Λ P F Λ P ~F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

197. O Passa F Λ P F Λ P ~F Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

198. O Passa F Λ P F Λ P ~F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

199. O Passa F Λ P F Λ P ~F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

200. O Passa F Λ P F Λ P ~F Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

201. O Passa F Λ P F Λ P ~F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

202. O Passa F Λ P F Λ P ~F Λ P Eb F Λ P Eb ~F Λ ~P Passb F

203. O Passa F Λ P F Λ P ~F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

204. O Passa F Λ P F Λ P ~F Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

205. O Passa F Λ P F Λ P ~F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

206. O Passa F Λ P F Λ P ~F Λ Eb F

207. O Passa F Λ P F Λ P ~F Λ O Passb F Λ O F

208. O Passa F Λ P F Λ P ~F Λ O Passb F Λ O ~F

209. O Passa F Λ P F Λ P ~F Λ O Passb F Λ P F Λ P ~F

210. O Passa F Λ P F Λ P ~F Λ O Eb ~F

211. O Ea ~F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ P(~F Λ ~ Eb ~F)

212. O Ea ~F Λ P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

213. O Ea ~F Λ P Eb F Λ P Eb ~F Λ P(F Λ ~Eb F) Λ ~ P(~F Λ ~Eb ~F)

214. O Ea ~F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

215. O Ea ~F Λ P Eb F Λ ~ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

216. O Ea ~F Λ P Eb F Λ ~ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

217. O Ea ~F Λ P Eb F Λ P Eb ~F Λ ~P Passb F

218. O Ea ~F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ P(~F Λ ~Eb ~F)

219. O Ea ~F Λ ~P Eb F Λ P Eb ~F Λ ~ P(F Λ ~ Eb F) Λ P(~F Λ ~ Eb ~F)

220. O Ea ~F Λ ~P Eb F Λ P Eb ~F Λ P(F Λ ~ Eb F) Λ ~ P(~F Λ ~ Eb ~F)

221. O Ea ~F Λ Eb F

222. O Ea ~F Λ O Passb F Λ O F

223. O Ea ~F Λ O Passb F Λ O ~F

224. O Ea ~F Λ O Passb F Λ P F Λ P ~F

225. O Ea ~F Λ O Eb ~F

Σε περιπτώσεις όπως αυτή, όπου ο αριθμός των πρακτόρων είναι μεγαλύτερος είτε ίσος του 2, η πλατφόρμα εμφανίζει τις 15 Κανονιστικές Θέσεις για κάθε πράκτορα ξεχωριστά και στη συνέχεια εμφανίζει τις Κανονιστικές Θέσεις για τους 2 πράκτορες ταυτόχρονα. Στο συγκεκριμένο παράδειγμα, για 2 πράκτορες a,b οι Κανονιστικές Θέσεις είναι: 15 * 15 = 225 Κανονιστικές Θέσεις.

4.2 Συμπεράσματα για την πλατφόρμα

Για να συνειδητοποιήσει κάποιος πλήρως την αναγκαιότητα ύπαρξης μιας πλατφόρμας που θα αυτοματοποιεί τη διαδικασία υπολογισμού των Κανονιστικών Θέσεων, αρκεί να διαβάσει το παράδειγμα 4.1.2 όπου έγινε ο υπολογισμός των 225 Κανονιστικών Θέσεων για 2 πράκτορες και ένα state of affairs. Για να γίνει αυτός ο υπολογισμός με το χέρι χρειάζεται τεράστιος χρόνος, ενώ με το πρόγραμμα αυτό χρειάστηκε μόλις ένα δευτερόλεπτο. Η πλατφόρμα έχει δημιουργηθεί για να υποστηρίζει μέχρι και n πράκτορες. Ωστόσο, όταν το n είναι μεγαλύτερο είτε ίσο του 5, ο χρόνος που απαιτείται αρχίζει να αυξάνεται δραματικά.

Πρακτικά, αυτό σημαίνει πως για περιπτώσεις όπου ο αριθμός των πρακτόρων είναι μεγάλος, απαιτείται ισχυρός επεξεργαστής, που θα μπορεί να ανταπεξέλθει στις απαιτήσεις που επιβάλλει η πολυπλοκότητα των υπολογισμών. Αυτό, καθιστά σχετικά δύσχρηστο το πρόγραμμα σε ένα μέσο " Personal Computer " και σε αυτές τις περιπτώσεις, απαιτείται η χρήση ενός ισχυρότερου υπολογιστικού συστήματος που θα μπορεί να ανταπεξέλθει στις αυξημένες απαιτήσεις της πλατφόρμας.

Τέλος, αξίζει να σημειωθεί πως ο αλγόριθμος αυτός δεν είναι ο βέλτιστος δυνατός καθώς, όπως έχει προτείνει και ο M Sergot, με τη κατάλληλη χρήση γραφικών ή με την έξυπνη χρήση υποσυνόλων, μπορεί να επιτευχθεί δραματική μείωση του αριθμού των Κανονιστικών Θέσεων. Ο αλγόριθμος αυτός ενώ υπολογίζει τον μικρότερο δυνατό αριθμό Κανονιστικών Θέσεων για ένα συγκεκριμένο αριθμό πρακτόρων, ανάλογα με τον σκοπό για τον οποίο χρησιμοποιείται, κάποιες από αυτές τις Κανονιστικές Θέσεις μπορεί να προσδίδουν άχρηστες για τον χρήστη πληροφορίες. Για παράδειγμα, αν το πρόγραμμα αυτό χρησιμοποιείται για τον υπολογισμό των Κανονιστικών Θέσεων που αφορούν ένα parking, θα ήταν πιο πρακτικό για τον θυρωρό να μπορεί να γνωρίζει ανά πάσα στιγμή τις Κανονιστικές Θέσεις των χρηστών που σχετίζονται άμεσα με κάποια συγκεκριμένη πληροφορία, παρά τις Κανονιστικές Θέσεις όλων των χρηστών του parking.

ΚΕΦΑΛΑΙΟ 5 – ΒΕΛΤΙΩΣΕΙΣ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ

Ο αλγόριθμος αυτός είναι εμπνευσμένος από το υπολογιστικό σύστημα Norman-G, το οποίο είναι ο μοναδικός τρόπος παραγωγής Κανονιστικών Θέσεων. Τα βασικά χαρακτηριστικά του Norman-G είναι τα εξής[Sergot, 2001] :

- Το συγκεκριμένο υπολογιστικό σύστημα χρησιμοποιεί το ακόλουθο set εκφράσεων για τη παραγωγή των Κανονιστικών Θέσεων:

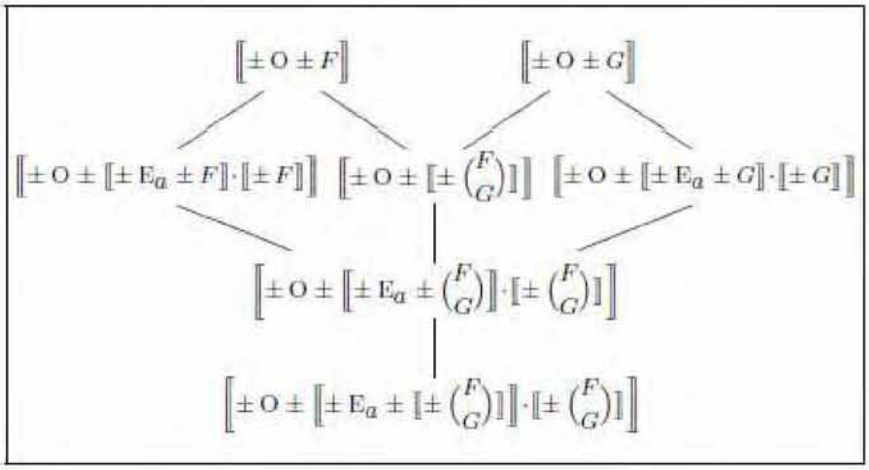
  $[\![\pm 0 \pm [\![\pm Ea \pm F]\!] * [\![\pm F]\!]]\!]$ / (A) όπου A κάποια έκφραση η οποία είναι λογικά συνεπής σχετικά με το λόγο για τον οποίο υπολογίζονται οι Κανονιστικές Θέσεις. Η χρήση μιας τέτοιας έκφρασης επιτρέπει τον επιμερισμό των Κανονιστικών Θέσεων έτσι ώστε ο χρήστης να λαμβάνει από αυτές μόνο τις πληροφορίες για τις οποίες ενδιαφέρεται. Αυτό μειώνει δραματικά τον αριθμό των Κανονιστικών Θέσεων που παράγονται, με αποτέλεσμα το πρόγραμμα να χρειάζεται μικρότερο χρόνο εκτέλεσης και λιγότερους πόρους.

Για παράδειγμα, έστω ένα parking το οποίο έχει ένα χρήστη a με 2 αυτοκίνητα α1, α2, ένα χρήστη b με ένα αυτοκίνητο b1 και έναν θυρωρό c. Έστω τα state of affairs p(a1), p(a2), p(b1) που δηλώνουν ότι το αντίστοιχο αυτοκίνητο είναι παρκαρισμένο στο parking. Έστω ότι για τη λειτουργία του parking είναι γνωστό ότι ο b απαγορεύεται να παρκάρει και ότι ο α επιτρέπεται να παρκάρει αλλά δεν είναι υποχρεωμένος να παρκάρει. Άρα, αν ήθελε ο θυρωρός να υπολογίσει τις Κανονιστικές Θέσεις για αυτή τη περίπτωση, αρκεί να υπολογίσει μόνο αυτές που είναι λογικά συνεπείς με την έκφραση A: –P(pb) Λ [P(pa1) Λ –P(pa1)] Λ [P(pa2) Λ –P(pa2)] αντί να υπολογίσει όλες τις Κανονιστικές Θέσεις για 3 πράκτορες και 3 state of affairs.

- Το συγκεκριμένο υπολογιστικό σύστημα χρησιμοποιεί γραφικά προκειμένου να κάνει πιο απλή και ταυτόχρονα πιο λεπτομερής τη διαδικασία δημιουργίας των Κανονιστικών Θέσεων. Αυτό επιτρέπει στο χρήστη να έχει ανά πάσα στιγμή πρόσβαση σε οποιαδήποτε υποσύνολο από Κανονιστικές Θέσεις που δημιουργείται με τον επιμερισμό της έκφρασης A.
- Το συγκεκριμένο υπολογιστικό σύστημα έχει τη δυνατότητα να υπολογίζει τις Κανονιστικές Θέσεις για n αριθμό πρακτόρων και m αριθμό state of affairs.

Οι 3 αυτοί λόγοι θα μπορούσαν να αποτελέσουν βασικές βελτιώσεις της πλατφόρμας. Για να γίνουν δυνατές, θα πρέπει να γίνουν αλλαγές στο κώδικα προκειμένου να υποστηρίζει η πλατφόρμα το set εκφράσεων που χρησιμοποιεί το Norman-G.

Επίσης, ο αλγόριθμος θα μπορούσε να υποστηρίξει περισσότερα state of affairs, ο βασικός λόγος όμως που δεν έγινε κάτι τέτοιο είναι ότι με την επιλογή πολλών state of affairs, αυξάνεται υπερβολικά ο αριθμός των Κανονιστικών Θέσεων, γεγονός που θα έκανε το πρόγραμμα σχετικά δύσκολο στη χρήση και αρκετά πιο αργό.



Σχήμα 6: Positions για ένα πράκτορα a και δύο state of affairs F, G, M. Sergot, 2001, page 29, Figure 5

Είναι χαρακτηριστικό πως, χωρίς την χρήση γραφικών και την τεχνική επιμερισμού με την έκφραση Α, για ένα παράδειγμα με 2 πράκτορες και 2 state of affairs ο χρήστης, αν μπορούσε να διαβάζει 10 Κανονιστικές Θέσεις το δευτερόλεπτο θα χρειαζόταν 58 δισεκατομμύρια χρόνια για να διαπεράσει όλη τη λίστα των Κανονιστικών Θέσεων.

Η σημαντικότερη ίσως βελτίωση που θα μπορούσε να γίνει στη πλατφόρμα είναι η δημιουργία μιας διεπαφής που θα προσφέρει τη δυνατότητα στο χρήση να επιλέγει ποιές Κανονιστικές Θέσεις θα ήθελε να είναι αποδεκτές στο σύστημά του και ποιές να απαγορεύονται. Ανάλογα με τις επιλογές του χρήστη, θα μπορούσε να δημιουργεί ένας αλγόριθμος που θα " κλαδεύει " αυτόματα τη λίστα με τις Κανονιστικές Θέσεις διαγράφοντας τις Κανονιστικές Θέσεις που απαγορεύονται ενώ ταυτόχρονα θα επισημαίνει στο χρήστη ποιές από τις Κανονιστικές Θέσεις που απομένουν είναι αποδεκτές.

Άλλη μία μεγάλη βελτίωση στο τρόπο λειτουργίας του προγράμματος θα ήταν μία διεπαφή που θα προσφέρει τη δυνατότητα στο χρήστη να κάνει ερωτήσεις στο σύστημα σχετικά με το αν μία Κανονιστική Θέση είναι επιτρεπτή ή όχι. Μια τέτοια λειτουργία, θα επέτρεπε στο χρήστη να γνωρίζει ανά πάσα στιγμή αν μια Κανονιστική Θέση είναι επιτρεπτή χωρίς να απαιτείται από αυτόν να διατρέξει τη τεράστια λίστα με τις Κανονιστικές Θέσεις, κάτι που θα έκανε το σύστημα πολύ πιο πρακτικό στη χρήση από ότι στη παρούσα κατάσταση που βρίσκεται.

Τέλος, η πλατφόρμα όπως έχει αναφερθεί και προηγουμένως, χρησιμοποιεί το set εκφράσεων του M. Sergot για την παραγωγή των Κανονιστικών Θέσεων. Έτσι, σημαντική βελτίωση για τον αλγόριθμο θα ήταν η προσθήκη επιπλέον κώδικα που θα επιτρέπει την χρήση διαφορετικών set εκφράσεων και μία διεπαφή που θα προσφέρει τη δυνατότητα στο χρήστη να επιλέγει με βάση ποιό set εκφράσεων θα ήθελε να γίνει η παραγωγή των Κανονιστικών Θέσεων.

ΚΕΦΑΛΑΙΟ 6 - ΒΙΒΛΙΟΓΡΑΦΙΑ

[Chellas, 1980] Brian F. Chellas. Modal Logic-An Introduction. Cambridge University Press, 1980.

[Hohfeld, 1913] W. N. Hohfeld. Some fundamental legal conceptions as applied in judicial reasoning. Yale Law Journal, 23, 1913. Reprinted with revisions as [Hohfeld, 1919 1923 1964] and [Hohfeld, 1978].

[Hohfeld, 1919 1923 1964] W. N. Hohfeld. (Revised version). In W. W. Cook, editor, Some Fundamental Legal Conceptions as Applied in Judicial Reasoning, and Other Legal Essays. Yale University Press, 1919, 1923, 1964.

[Hohfeld, 1978] W. N. Hohfeld. (Revised version). In W. C. Wheeler, editor, Some Fundamental Legal Conceptions as Applied in Judicial Reasoning, and Other Legal Essays. Greenwood Press, 1978.

[Jones and Sergot, 1992] A. J. I. Jones and M. J. Sergot. Formal specification of security requirements using the Theory of Normative Positions. In Y. Deswarte, G. Eizenberg, and J.-J. Quisquater, editors, Computer Security-ESORICS 92, LNCS 648, pages 103-121. Springer-Verlag, Berlin Heidelberg, 1992.

[Jones and Sergot, 1993] A. J. I. Jones and M. J. Sergot. On the Characterisation of Law and Computer Systems: The Normative Systems Perspective. In John-

Jules Ch. Meyer and Roel J. Wieringa, editors, Deontic Logic in Computer Science: Normative System Specification, chapter 12, pages 275-307. John Wiley & Sons, Chichester, England, 1993.

[Kanger and Kanger, 1966] S. Kanger and H. Kanger. Rights and Parliamentarism. Theoria, 32:85-115, 1966.

[Kanger, 1971] S. Kanger. New foundations for ethical theory. In R. Hilpinen, editor, Deontic Logic: Introductory and Systematic Readings, pages 36-58. D. Reidel, Dordrecht, 1971. Originally published as Technical Report, Stockholm University, 1957.

[Kanger, 1972] S. Kanger. Law and Logic. Theoria, 38:105-132, 1972. Normative Positions 53

[Kanger, 1985] S. Kanger. On Realization of Human Rights. In G. Holmstrom and A. J. I. Jones, editors, Action, Logic and Social Theory. Acta Philosophica Fennica, Vol. 38, 1985.

[Lee, 1988] R. M. Lee. Bureaucracies as deontic systems. ACM Transactions on Information Systems, 6(2):87-108, 1988.

[Lindahl, 1977] Lars Lindahl. Position and Change-A Study in Law and Logic. Number 112 in Synthese Library. D. Reidel, Dordrecht, 1977.

[Lindahl, 1994] Lars Lindahl. Stig Kanger's Theory of Rights. In D Prawitz, B Skyrms, and D Westerstahl, editors, Logic, Methodology and Philosophy of Science IX, pages 889-911, New York, 1994. Elsevier Science Publishers B.V.

[Makinson, 1986] David Makinson. On the formal representation of rights relations. Journal of Philosophical Logic, 15:403-425, 1986.

[Porn, 1977] Ingmar Porn. Action Theory and Social Science: Some Formal Models. Number 120 in Synthese Library. D. Reidel, Dordrecht, 1977.

[Sergot, 2001] M. J. Sergot. A computational theory of normative positions. ACM Transactions on Computational Logic, 2(4):581-622, October 2001.

.