ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΜΕΤΑΠΤΥΧΙΑΚΟ ΜΑΘΗΜΑΤΙΚΟΥ-ΜΗΧΑΝΙΚΩΝ Η/Υ

ΕΥΦΥΕΣ ΣΥΣΤΗΜΑ ΧΟΡΗΓΗΣΗΣ ΑΣΦΑΛΕΙΩΝ

Διπλωματική Εργασία της Δασκαλάκη Ευφροσύνης,

Α. Μ. 140

Επιβλέπων Καθηγητής,

Ιωάννης Χατζηλυγερούδης

Επίκουρος Καθηγητής Πανεπιστημίου Πατρών

Πάτρα, 2009

Περιεχόμενα

Ευχαριστίες

Στο σημείο αυτό, θα ήθελα να ευχαριστήσω τον επιβλέποντα της παρούσας Μεταπτυχιακής Εργασίας, κο. Χατζηλυγερούδη Ιωάννη, Επίκουρο Καθηγητή του Πανεπιστημίου Πατρών, για τη συνεχή καθοδήγησή του και τη βοήθειά του στην εκπόνηση της εργασίας αυτής.

Ακόμα, θα ήθελα να ευχαριστήσω την οικογένειά μου που με στηρίζει όλα αυτά τα χρόνια, και το σύζυγό μου για όλη την ψυχολογική υποστήριξη που μου έχουν παράσχει.

Ιδιαίτερες ευχαριστίες οφείλω στην μητέρα μου, χωρίς τη συμβολή της οποίας δε θα μπορούσα να ολοκληρώσω αυτήν την εργασία, καθώς μου παρείχε όλες τις τεχνικές πληροφορίες και τα δεδομένα που χρειάστηκα για να αναπτύξω το πρόβλημα της Ασφαλισιμότητας.

Εισαγωγή

Στην εργασία που ακολουθεί, ασχολούμαστε με την εφαρμογή μεθόδων Τεχνητής Νοημοσύνης σε ένα πραγματικό πρόβλημα, που αναφέρεται στην διάγνωση του βαθμού *ασφαλισιμότητας* ενός πελάτη μιας ασφαλιστικής εταιρείας. Η ανάγκη για την εφαρμογή αυτή προέκυψε από το γεγονός ότι πολλές φορές ο εμπειρογνώμονας της εταιρείας δεν είναι διαθέσιμος, αλλά και όταν είναι, χρειάζεται ένα συμβουλευτικό πρόγραμμα.

Πιο συγκεκριμένα, για τη λύση του προβλήματος χρησιμοποιούνται: α) ένα ασαφές έμπειρο σύστημα υλοποιημένο με τη βοήθεια του εργαλείου FuzzyCLIPS [1], β) ένα έμπειρο σύστημα που χρησιμοποιεί κανόνες με συντελεστές βεβαιότητας τύπου MYCIN, γ) ένα έμπειρο σύστημα που χρησιμοποιεί κανόνες με συντελεστές βεβαιότητας τύπου weighted [6], υλοποιημένα και τα δύο με βάση το εργαλείο CLIPS [2] και δ) ένα νευρωνικό δίκτυο υλοποιημένο με βάση το εργαλείο WEKA [10]. Στο τέλος συγκρίνουμε τα παραπάνω συστήματα με βάση κάποιες μετρικές.

Πριν να ξεκινήσουμε την ανάλυση του προβλήματός μας και των υλοποιήσεων των παραπάνω συστημάτων, αναλύουμε λίγο παραπάνω τους όρους και τα εργαλεία που ήδη αναφέραμε, δίνοντας περισσότερες πληροφορίες για την προέλευση τους, τα χαρακτηριστικά τους, τη χρησιμότητά τους, κτλ.

Έτσι, στο πρώτο κεφάλαιο δίνουμε περισσότερα στοιχεία για τον τομέα της Τεχνητής Νοημοσύνης και πώς αυτός έχει εξελιχτεί στις τελευταίες δεκαετίες. Στο δεύτερο κεφάλαιο, αναλύουμε τη συσχέτιση των Έμπειρων Συστημάτων με την Τεχνητή Νοημοσύνη, τα χαρακτηριστικά τους, τη δομή τους, τα πλεονεκτήματα και μειονεκτήματά τους.

Στη συνέχεια, και στα κεφάλαια 3, 4 και 5, αναλύουμε τα τρία εργαλεία που θα χρησιμοποιήσουμε και τις δυνατότητες αυτών. Κι αφού δώσουμε περισσότερες πληροφορίες για το πρόβλημα της ‘Ασφαλισιμότητας’ και τον τρόπο που το αντιμετωπίζουμε στο κεφάλαιο 6, στα επόμενα τρία κεφάλαια (κεφάλαια 7, 8, και 9) γίνεται παρουσίαση των παραπάνω ευφυών συστημάτων και των αποτελεσμάτων τους σε συγκεκριμένο σύνολο δεδομένων.

Τέλος, στο κεφάλαιο 10 προχωράμε σε σύγκριση και σχολιασμό των τιμών των μετρικών που προέκυψαν από τις προηγούμενες εφαρμογές, και εξαγωγή των συμπερασμάτων της σύγκρισης.

1. Τεχνητή Νοημοσύνη (Artificial Intelligence)

1.1 Γενικά

Η Τεχνητή Νοημοσύνη (ΤΝ) είναι ο τομέας της πληροφορικής που εστιάζει στη δημιουργία των μηχανών, που μπορούν να συμμετέχουν στις συμπεριφορές που οι άνθρωποι θεωρούν ευφυής. Η δυνατότητα να δημιουργηθούν οι ευφυείς μηχανές έχει κεντρίσει τους ανθρώπους από την αρχή της βιομηχανικής εποχής, και μετά από 50 έτη έρευνας, το όνειρο των έξυπνων μηχανών γίνεται βαθμιαία μια πραγματικότητα [1], [3].

Γινόμαστε όλο και περισσότερο πιο εξαρτώμενοι από τις ευφυείς μηχανές. Οι ευφυείς μηχανές βρίσκονται παντού γύρω μας στην εποχή μας. Τουλάχιστον μια φορά την ημέρα, οι περισσότεροι άνθρωποι θα χρησιμοποιήσουν μια μηχανή αυτόματης ανάληψης μετρητών (ATM), ένα αυτοκίνητο, ή ένα φούρνο μικροκυμάτων. Ο κατάλογος των συστημάτων που ελέγχονται από έναν υπολογιστή είναι απέραντος. Οι προσωπικοί Η/Υ έχουν γίνει απαραίτητες οικιακές συσκευές, και όλο και περισσότεροι άνθρωποι έρχονται σε επαφή με τους υπολογιστές σε προσωπικό επίπεδο. Η ζήτηση αυξάνεται, και εκτιμάται ότι αυτή η τεχνολογία πρέπει όχι μόνο να εκπληρώσει τις απαιτήσεις μας αλλά να προλαμβάνει τις ανάγκες μας. Ο καταναλωτής *θέλει* αυτήν την τεχνολογία να γίνει 'ευφυής' και οι υπεύθυνοι για την ανάπτυξη προσπαθούν σκληρά να επιτύχουν αυτόν τον στόχο.

Δεδομένου ότι η γνώση του ατόμου εξελίσσεται, γίνεται ακόμα πιο επιτακτική η ανάγκη να ενσωματωθεί αυτή η ικανότητα και σε μια μηχανή. Δεν είναι απίθανο ότι μια ημέρα θα είναι εφικτό να χτιστούν τεχνητές μηχανές των οποίων νοημοσύνη να φτάνει, και ενδεχομένως ακόμη και να υπερβαίνει, αυτή των ανθρώπων. Η μελέτη της τεχνητής νοημοσύνης έχει παράσχει τις καλύτερες τεχνικές προγραμματισμού για την κατασκευή εξυπνότερων συστημάτων ηλεκτρονικών υπολογιστών. Το ερώτημα όμως είναι: Μπορούν οι υπολογιστές να γίνουν ευφυείς; Είναι αυτό πραγματικά εφικτό; Και αν είναι, με ποιόν τρόπο;

Η τεχνητή νοημοσύνη είναι ένας σχετικά νέος κλάδος της πληροφορικής, αν και η επιτακτική ανάγκη της ύπαρξης 'έξυπνων' μηχανών στη ζωή μας, έχει δώσει μεγάλη ώθηση στην εξέλιξη και πρακτική εφαρμογή της στην καθημερινότητά μας. Παρακάτω αναφέρονται εν

συντομία η ιστορία της εξέλιξής της τις τελευταίες δεκαετίες:

1.2 Ιστορία της Τεχνητής Νοημοσύνης

Στο [5] αναφέρονται οι παρακάτω σημαντικοί σταθμοί στην ιστορία της Τεχνητής Νοημοσύνης:

1936 Ο Alan Turing λαμβάνει υπόψη του τη λειτουργία του ανθρώπινου εγκεφάλου για την εφαρμογή της στον προγραμματισμό.

1943 Οι Warren McCulloch και Walter Pitts κατασκευάζουν ένα απλό νευρωνικό δίκτυο.

1947 Ο Arthur Samuel της IBM ξεκινάει να δουλεύει ένα πρόγραμμα για να παίζει Σκάκι που έχει την ικανότητα να μαθαίνει από τα λάθη του.

1950 Ο Alan Turing περιγράφει τη δοκιμή Turing, που επιδιώκει να εξετάσει την ικανότητα μιας μηχανής να συμμετάσχει απρόσκοπτα σε μια ανθρώπινη συνομιλία.

1951 Τα πρώτα προγράμματα ΤΝ γράφονται για τον υπολογιστή Ferranti Mark I στο Πανεπιστήμιο του Μάντσεστερ: ένα πρόγραμμα που παίζει ντάμα από τον Cristopher Strakli και ένα που παίζει σκάκι από τον Dietrich Prinz.

1956 Ο John Mccarthy πλάθει τον όρο «Τεχνητή Νοημοσύνη» ως κύριο θέμα της διάσκεψης του Dartmouth.

1958 Ο John Mccarthy εφευρίσκει τη γλώσσα προγραμματισμού Lisp.

1965 Ο Edward Feigenbaum ξεκινά το Dendral, μια δεκαετή προσπάθεια ανάπτυξης λογισμικού που θα συμπεράνει τη μοριακή δομή οργανικών ενώσεων χρησιμοποιώντας ενδείξεις επιστημονικών οργάνων. Ήταν το πρώτο έμπειρο σύστημα (expert system).

1966 Ιδρύεται το Εργαστήριο Μηχανικής Νοημοσύνης στο Εδιμβούργο –

το πρώτο από μια σημαντική σειρά εγκαταστάσεων που οργανώνονται από τον Donald Mitsu και άλλους.

1970 Αναπτύσσεται το Planner και χρησιμοποιείται στο SHRDLU, μια εντυπωσιακή επίδειξη αλληλεπίδρασης μεταξύ ανθρώπου και υπολογιστή.

1971 Ξεκινά η εργασία πάνω στο σύστημα αυτόματης απόδειξης θεωρημάτων Boyer-Moore στο Εδιμβούργο.

1972 Η γλώσσα προγραμματισμού Prolog αναπτύσσεται από τον Alain Colmerauer.

1973 Ρομπότ συναρμολόγησης «Freddy» στο Εδιμβούργο: ένα ευπροσάρμοστο σύστημα συναρμολόγησης που ελέγχεται από υπολογιστές.

1974 Ο Tedd Shortliffe γράφει τη διατριβή του για το πρόγραμμα MYCIN (Στάνφορντ), το οποίο κατέδειξε μια πολύ πρακτική προσέγγιση στην ιατρική διάγνωση που βασίζεται σε κανόνες, ενώ λειτουργεί ακόμα και με παρουσία αβεβαιότητας. Αν και δανείστηκε από το DENDRAL, οι δικές του συνεισφορές επηρέασαν έντονα το μέλλον των έμπειρων συστημάτων, ένα μέλλον με πολλαπλές εμπορικές εφαρμογές.

1991 Η εφαρμογή σχεδίασης ενεργειών DART χρησιμοποιείται αποτελεσματικά στον Α' Πόλεμο του Κόλπου και ανταμείβει 30 χρόνια έρευνας στην ΤΝ του Αμερικανικού Στρατού.

1994 Οι Dickmanns και Daimler- Benz οδηγούν περισσότερο από 1000 km σε μια εθνική οδό του Παρισιού υπό συνθήκες βαρείας κυκλοφορίας και σε ταχύτητες ως και 130 km/ώρα. Επιδεικνύουν αυτόνομη οδήγηση σε ελεύθερες παρόδους, οδήγηση σε συνοδεία, αλλαγή παρόδων και αυτόματη προσπέραση άλλων οχημάτων.

1997 Ο υπολογιστής Deep Blue της IBM κερδίζει τον παγκόσμιο πρωταθλητή σκακιού Garry Kasparov.

1998 Κυκλοφορεί ο Ferby της Tiger Electronics και γίνεται η πρώτη επιτυχημένη εμφάνιση ΤΝ σε οικιακό περιβάλλον.

1999 Η Sony λανσάρει το TNBO, που είναι ένα από τα πρώτα αυτόνομα κατοικίδια ΤΝ.

2004 Η DARPA ξεκινά το πρόγραμμα DARPA Grand Challenge («Μεγάλη Πρόκληση DARPA»), που προκαλεί τους συμμετέχοντες να δημιουργήσουν αυτόνομα οχήματα για ένα χρηματικό βραβείο.

2. Έμπειρα Συστήματα και Τεχνητή Νοημοσύνη

2.1 Γενικά

Τα Έμπειρα Συστήματα (Expert Systems),ή αλλιώς Συστήματα Βασισμένα στη Γνώση (Knowledge-Based Systems) ή απλά Συστήματα Γνώσης (Knowledge Systems), είναι υπολογιστικά προγράμματα που χρησιμοποιούν τη γνώση για να μας βοηθήσουν να εκτελέσουμε μια μεγάλη ποικιλία ενεργειών, συμπεριλαμβανομένων διαγνώσεων, προγραμματισμού, οργάνωσης και σχεδιασμού.

Τα έμπειρα συστήματα ΕΣ είναι υπολογιστικά προγράμματα που προέρχονται από τον κλάδο της επιστήμης υπολογιστών, που αναλύσαμε νωρίτερα, την Τεχνητή Νοημοσύνη (ΤΝ). Ο επιστημονικός στόχος της ΤΝ είναι να γίνει κατανοητή η νοημοσύνη με την κατασκευή υπολογιστικών προγραμμάτων που εκθέτουν την ευφυή συμπεριφορά. Ενδιαφέρεται για τις έννοιες και τις μεθόδους συμβολικού συμπεράσματος, ή συλλογισμού, από έναν υπολογιστή, και πώς η γνώση που χρησιμοποιείται για να διεξαγάγει εκείνα τα συμπεράσματα θα αντιπροσωπευθεί μέσα στη μηχανή [1].

Όπως είναι φυσικό, ο όρος 'νοημοσύνη' καλύπτει πολλές γνωστικές δεξιότητες, συμπεριλαμβανομένης της ικανότητας της επίλυσης προβλημάτων, την εκμάθηση, και την κατανόηση της γλώσσας. Η ΤΝ περικλείει όλα τα προηγούμενα, αλλά η μεγαλύτερη πρόοδος μέχρι σήμερα στην ΤΝ έχει σημειωθεί στον τομέα της επίλυσης προβλήματος - έννοιες και μέθοδοι για την κατασκευή προγραμμάτων που μελετούν τα προβλήματα παρά απλά υπολογίζουν μια λύση.

Χιλιάδες συστήματα χρησιμοποιούνται καθημερινά παγκοσμίως, και επεκτείνεται το φάσμα των εφαρμογών στις επιχειρήσεις, τη βιομηχανία και την κυβέρνηση. Μεγάλο οικονομικό κέρδος έχει επιτευχθεί σε πολλές διαστάσεις: επιτάχυνση της επαγγελματικής (και ημί-επαγγελματικής) εργασίας, εσωτερική εξοικονόμηση κόστους στις εγχειρήσεις, βελτιωμένες ποιότητα και συνέπεια των λήψεων αποφάσεων, αυξημένο εισόδημα από τα νέα προϊόντα και τις υπηρεσίες, απόκτηση οργανωτικής τεχνογνωσίας, βελτιώσεις στον τον τρόπο που μια επιχείρηση λειτουργεί, καλύτερη διαχείριση κρίσεων και υποκίνηση της καινοτομίας.

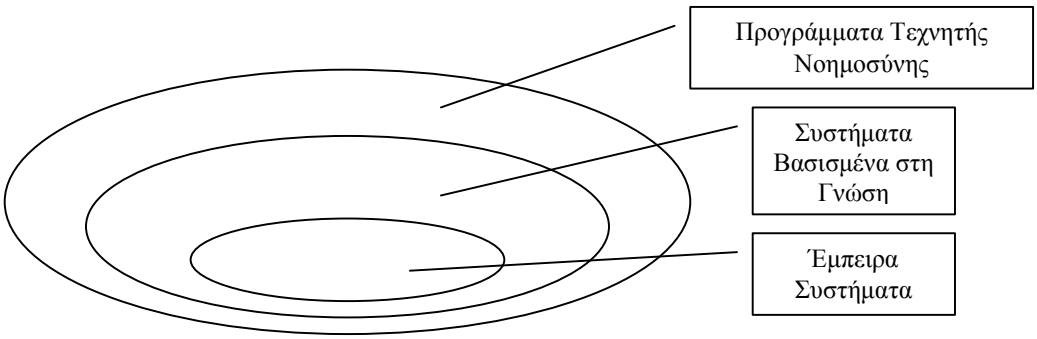
Τα έμπειρα συστήματα δηλαδή, είναι μια περιοχή της ΤΝ που εξερευνά πώς να απεικονίσει σε ένα υπολογιστικό πρόγραμμα την πείρα ενός ανθρώπινου εμπειρογνώμονα. Παραδείγματος χάριν, είναι πιθανό να απεικονισθεί η γνώση ενός ιατρικού ειδικού στις διαγνώσεις ή ενός κύριου προσώπου επισκευής υπολογιστών;

Έχουμε εξοικειωθεί με την ιδέα ότι ένα μεγάλο μέρος της γνώσης ενός εμπειρογνώμονα μπορεί να μεταφερθεί σε ένα βιβλίο. Το βιβλίο μπορεί να σχεδιαστεί για να βοηθήσει έναν άνθρωπο να μάθει μερικές από τη γνώση του συγγραφέα. Ή, μπορεί να περιέχει βήμα- βήμα διαδικασίες που, εάν ακολουθούνται προσεκτικά, θα λύσουν συγκεκριμένα προβλήματα ή θα ολοκληρώσουν συγκεκριμένους στόχους που μέχρι εκείνη τη στιγμή πραγματοποιούνταν από έναν ανθρώπινο εμπειρογνώμονα.

Τα ΕΣ διακρίνονται από τα συμβατικά υπολογιστικά προγράμματα σε δύο βασικές κατηγορίες (Barr, Cohen et al. 1989):

- ΕΣ που ασχολούνται με συγκεκριμένα πεδία γνώσης, τα οποία είναι τόσο συμβολικά όσο και αριθμητικά,
- ΕΣ που χρησιμοποιούν συγκεκριμένα πεδία μεθόδων που είναι τόσο ευρετικά (δηλ. πιθανά) όσο και αλγοριθμικά (δηλ. βέβαια)

Τα ΕΣ αποτελούν την πιο επιτυχή εμπορική εφαρμογή της έρευνας στην Τεχνητή Νοημοσύνη.



Σχήμα 1. Τα έμπειρα συστήματα αποτελούν συστήματα βασισμένα στη γνώση για τα προγράμματα της Τεχνητής Νοημοσύνης.

2.2 Σημαντικά Συστατικά των Έμπειρων Συστημάτων

Η διεπαφή με τον χρήστη

Η διεπαφή με τον χρήστη είναι ο τρόπος επικοινωνίας μεταξύ ενός χρήστη και των διαδικασιών επίλυσης προβλήματος έμπειρων συστημάτων. Ένα καλό έμπειρο σύστημα δεν είναι πολύ χρήσιμο εκτός αν έχει μια αποτελεσματική διεπαφή. Πρέπει να είναι σε θέση να δεχτεί τις ερωτήσεις ή τις οδηγίες στη μορφή που ο χρήστης τις εισάγει, και να τις μεταφράζει στις οδηγίες εργασίας για το υπόλοιπο του συστήματος. Πρέπει επίσης να είναι σε θέση να μεταφράσει τις απαντήσεις που παράγονται από το σύστημα, σε μια μορφή που ο χρήστης μπορεί να καταλάβει. Ιδιαίτερη προσοχή πρέπει να δοθεί στο σχεδιασμό της οθόνης προκειμένου να γίνει το έμπειρο σύστημα φιλικό προς το χρήστη.

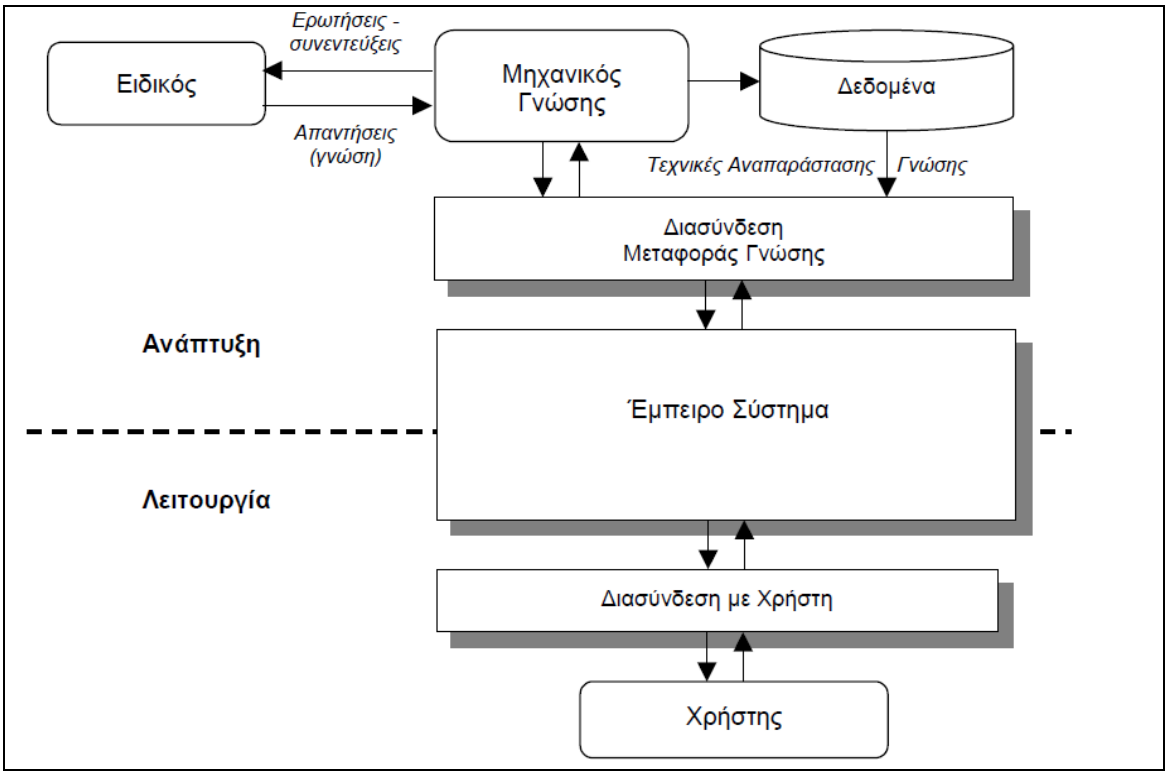
Η βάση γνώσεων

Η βάση γνώσεων αποθηκεύει όλα τα γεγονότα και τους κανόνες για ένα συγκεκριμένο μέρος του προβλήματος. Καθιστά τα γεγονότα διαθέσιμα στους κανόνες συμπερασμάτων σε μια μορφή που μπορούν να τα χρησιμοποιήσουν. Τα γεγονότα μπορούν να είναι υπό μορφή βασικών πληροφοριών που υπάρχουν στο σύστημα ή τα γεγονότα που εισάγονται από το χρήστη κατά τη διάρκεια της διεπαφής. Οι κανόνες περιλαμβάνουν και τους κανόνες παραγωγής που ισχύουν για το πεδίο του έμπειρου συστήματος και υποθέσεις (heuristics) που παρέχονται από τον ειδικό, προκειμένου το σύστημα να βρει τις λύσεις αποτελεσματικότερα με τη λήψη συντομότερων διαδρομών.

Το περίβλημα ή η μηχανή συμπεράσματος

Η μηχανή συμπεράσματος είναι το πρόγραμμα που εντοπίζει την κατάλληλη γνώση στη βάση γνώσεων, και συμπεραίνει τη νέα γνώση με την εφαρμογή των λογικών στρατηγικών επεξεργασίας και επίλυσης προβλήματος.

Από το Σχήμα 1 φαίνονται αυτά τα τρία βασικά συστατικά ενός έμπειρου συστήματος.



Σχήμα 2. Σχηματική αναπαράσταση έμπειρου συστήματος.

2.3 Τύποι Έμπειρων Συστημάτων

Υπάρχουν πολλοί διαφορετικοί τύποι έμπειρων συστημάτων. Ο ακόλουθος κατάλογος περιγράφει τους διάφορους τύπους.

Διάγνωσης (Diagnosis). Οι τύποι διαγνωστικών έμπειρων συστημάτων χρησιμοποιούνται για να συστήσουν τις θεραπείες στις ασθένειες, να ανιχνεύσουν λάθη τα ηλεκτρονικά ή μηχανικά προβλήματα ή ως εργαλεία διόρθωσης.

Επισκευής (Repair). Τα έμπειρα συστήματα που καθορίζουν τις στρατηγικές επισκευής είναι επίσης πολύ κοινά. Εκτός του να προχωρήσουν στη διάγνωση του προβλήματος, μπορούν να προτείνουν και ένα σχέδιο για την αντιμετώπισή του. Το σχέδιο αντιμετώπισης περιέχει συνήθως μια δομή σχεδιασμού και κάποια δομή ελέγχου, για να επικυρώσει τη διαδικασία επισκευής. Τέτοια συστήματα έχουν χρησιμοποιηθεί στον τομέα επισκευών αυτοκίνητων και σε άλλα παρόμοια προβλήματα.

Οδηγιών (Instruction). Τα εκπαιδευτικά έμπειρα συστήματα έχουν χρησιμοποιηθεί για την εξατομικευμένη εκπαίδευση ή την καθοδήγηση σε έναν συγκεκριμένο τομέα. Το σύστημα παρουσιάζει το υλικό εκμάθησης, το οποίο καθορίζεται από την αξιολόγησή της ικανότητας και της τρέχουσας γνώσης του χρήστη- σπουδαστή και καταγράφει την πρόοδό του, αλλάζοντας την ακολουθία των βημάτων ανάλογα με την πρόοδο αυτή.

Ερμηνείας (Interpretation). Τα ερμηνευτικά έμπειρα συστήματα έχουν τη δυνατότητα να αναλύσουν τα στοιχεία για να καθορίσουν τη σημασία ή τη χρησιμότητά του. Η βάση γνώσεων περιέχει συχνά μοντέλα καταστάσεων πραγματικών κόσμων τα οποία και συγκρίνει με τα στοιχεία της. Χρησιμοποιούνται συχνά στην εξερεύνηση μεταλλευμάτων, αποθεμάτων αερίου και πετρελαίου, καθώς επίσης και στην επιτήρηση, την ανάλυση εικόνας και την κατανόηση ομιλίας.

Πρόβλεψης (Prediction). Τα προφητικά έμπειρα συστήματα χρησιμοποιούνται ως μέθοδος «μαντείας» των πιθανών εκβάσεων των παρατηρηθεισών καταστάσεων, παρέχοντας συνήθως έναν παράγοντα πιθανότητας. Αυτό χρησιμοποιείται συχνά στην καιρική πρόβλεψη.

Σχεδίου και προγραμματισμού (Design and Planning). Επιτρέπει στους εμπειρογνώμονες να αναπτύξει γρήγορα λύσεις που εξοικονομούν χρόνο. Αυτά τα συστήματα δεν αντικαθιστούν τους εμπειρογνώμονες, αλλά ενεργούν ως εργαλείο για την εκτέλεση εργασιών όπως η κοστολόγηση, ο σχεδιασμός κτηρίων, η παραγγελία υλικών και το σχεδιασμό περιοδικών.

Παρακολούθησης καταστάσεων και ελέγχου (Monitoring and Control). Σε ορισμένες εφαρμογές τα έμπειρα συστήματα μπορούν να σχεδιαστούν για να επιτηρήσουν τις διαδικασίες και να ελέγξουν ορισμένες λειτουργίες. Αυτά τα συστήματα είναι ιδιαίτερα χρήσιμα όπου η ταχύτητα της απόφασης είναι ζωτικής σημασίας, παραδείγματος χάριν στην βιομηχανία πυρηνικής ενέργειας, στον έλεγχο εναέριας κυκλοφορίας και

στο χρηματιστήριο.

Ταξινόμησης/ Προσδιορισμού (Classification/ Identification). Αυτά τα συστήματα βοηθούν να ταξινομήσουν τους στόχους στο σύστημα από τον προσδιορισμό των διάφορων χαρακτηριστικών γνωρισμάτων (φυσικών και μη-φυσικών). Παραδείγματος χάριν, οι διάφοροι τύποι ζώων είναι ταξινομημένοι σύμφωνα με χαρακτηριστικά όπως ο βιότοπος, οι πληροφορίες σίτισης, το χρώμα, οι πληροφορίες αναπαραγωγής, το σχετικό μέγεθος κ.λπ. Αυτά τα συστήματα μπορούν να χρησιμοποιηθούν από τους παρατηρητές πουλιών, τους λάτρεις της αλιείας, τα καταφύγια διάσωσης ζώων (για να ταιριάξουν με τα ζώα τους ενδεχόμενους ιδιοκτήτες), κ.τ.λ. .

2.4 Δομικές Μονάδες των Έμπειρων Συστημάτων

Κάθε έμπειρο σύστημα αποτελείται από δύο κύρια μέρη: τη βάση γνώσεων και το συλλογισμό, ή συμπέρασμα, ή μηχανή.

Η βάση γνώσεων των έμπειρων συστημάτων περιέχει την πραγματική και την ευρετική (δυναμική) γνώση.

Η πραγματική γνώση είναι εκείνη η γνώση του πεδίου εργασιών που μοιράζεται ευρέως, μπορεί να αντληθεί συνήθως από τα εγχειρίδια ή τα περιοδικά, και είναι αποδεκτή συνήθως από τους ειδικούς στον συγκεκριμένο τομέα.

Η ευρετική γνώση είναι λιγότερο αυστηρή, πιο εμπειρική, και αποτελεί περισσότερο ‘μαντεία’ της απόδοσης. Σε αντίθεση με την πραγματική γνώση, η ευρετική γνώση είναι κατά ένα μεγάλο μέρος υποκειμενική. Είναι η γνώση της ορθής πρακτικής, της καλής κρίσης, και του εύλογου συλλογισμού στον κάθε τομέα. Είναι η γνώση που κρύβεται κάτω από την "τέχνη" της σωστής ‘αυθαίρετης’ επιλογής πιθανής λύσης.

Η αναπαράσταση της γνώσης τυποποιεί και οργανώνει τη γνώση. Μια ευρέως χρησιμοποιημένη αναπαράσταση είναι ο κανόνας παραγωγής, ή απλά κανόνας. Ένας κανόνας αποτελείται από ένα IF μέρος και ένα THEN

μέρος (ή αλλιώς συνθήκη και αποτέλεσμα). Οι συνθήκες IF αποτελούν ένα σύνολο συνθηκών με μία λογική σύνθεση. Το κομμάτι της γνώσης που αντιπροσωπεύεται από τον κανόνα παραγωγής είναι σχετικό με τη γραμμή αιτιολόγησης που θα αναπτυχθεί εάν το IF μέρος του κανόνα ικανοποιείται, συνεπώς, το THEN μέρος μπορεί να ολοκληρωθεί, ή να διενεργηθούν οι υπολογισμοί για την επίλυση του προβλήματός. Τα έμπειρα συστήματα των οποίων η γνώση αντιπροσωπεύεται με μορφή κανόνα, καλούνται συστήματα βασισμένα στους κανόνες (rule-based systems).

Μια άλλη ευρέως χρησιμοποιημένη αναπαράσταση της γνώσης, αποκαλούμενη ως μονάδα (επίσης γνωστή ως πλαίσιο, σχήμα, ή δομή καταλόγων) είναι βασισμένη σε μια παθητικότερη άποψη της γνώσης. Η μονάδα είναι μια συνάθροιση της συσχετιζόμενης συμβολικής γνώσης για ένα αντικείμενο που αντιπροσωπεύεται. Χαρακτηριστικά, μια μονάδα αποτελείται από έναν κατάλογο ιδιοτήτων του αντικειμένου και των σχετικών τιμών του για τις ιδιότητες που έχει.

Δεδομένου ότι κάθε πεδίο εργασιών αποτελείται από πολλά αντικείμενα που εμπλέκονται στις διάφορες σχέσεις, οι ιδιότητες μπορούν επίσης να χρησιμοποιηθούν για να διευκρινίσουν τις σχέσεις, και οι τιμές αυτών των ιδιοτήτων είναι τα ονόματα άλλων μονάδων που συνδέονται σύμφωνα με τις σχέσεις. Μια μονάδα μπορεί επίσης να αντιπροσωπεύσει τη γνώση που αποτελεί μια "ειδική περίπτωση" μιας άλλης μονάδας, ή μερικές μονάδες μπορεί να είναι "κομμάτια" μιας άλλης μονάδας.

Το μοντέλο επίλυσης προβλήματος, ή το παράδειγμα, οργανώνει και ελέγχει τα βήματα που ακολουθούνται για να λύσουν το πρόβλημα. Ένα κοινό αλλά ισχυρό παράδειγμα περιλαμβάνει τους αλυσιδωτούς IF- THEN κανόνες για να διαμορφωθεί μια γραμμή συλλογισμού. Εάν η αλυσίδα κανόνων αρχίζει από ένα σύνολο συνθηκών και κινείται προς κάποιο συμπέρασμα, η μέθοδος καλείται εμπρόσθια αλυσιδωτή δομή (forward chaining). Εάν το συμπέρασμα είναι γνωστό (παραδείγματος χάριν, ένας στόχος που επιθυμούμε να επιεύξουμε) αλλά η διαδρομή ως εκείνο το συμπέρασμα δεν είναι γνωστή, τότε ακολουθείται η αντίστροφη διαδικασία, και η μέθοδος καλείται οπίσθια αλυσιδωτή δομή (backward chaining).

Αυτές οι μέθοδοι επίλυσης προβλημάτων εφαρμόζονται μέσω δομών προγράμματος αποκαλούμενες μηχανές συμπεράσματος ή διαδικασίες συμπεράσματος που χειρίζονται και χρησιμοποιούν τη γνώση στη βάση γνώσεων για να διαμορφώσουν μια γραμμή συλλογισμού.

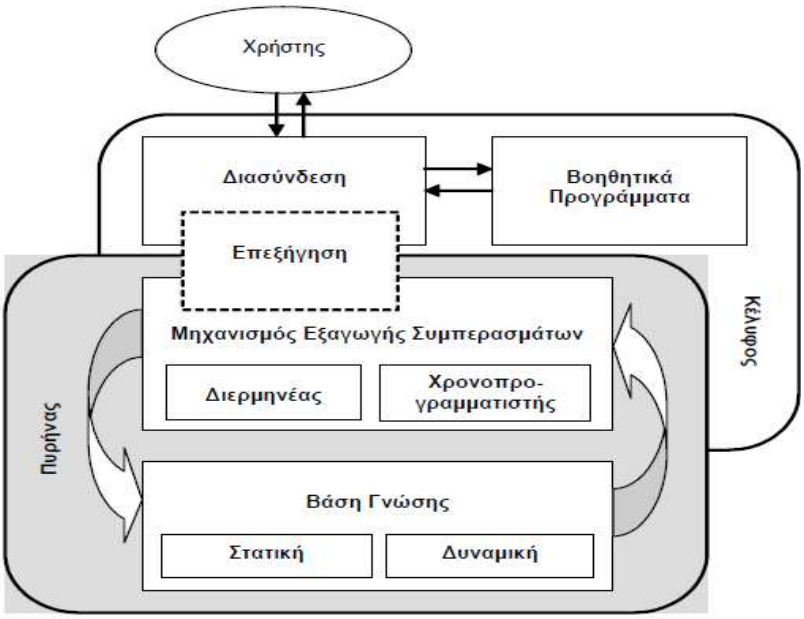
2.5 Αρχιτεκτονική Έμπειρων Συστημάτων

Σήμερα υπάρχουν δύο τρόποι να κατασκευαστεί ένα έμπειρο σύστημα. Μπορεί να κατασκευαστεί από την αρχή, ή χρησιμοποιώντας ένα κομμάτι του λογισμικού ανάπτυξης γνωστό ως " εργαλείο" (tool) ή ένα "κέλυφος" (shell).

Αν και υπάρχουν διαφορετικές μορφές και μέθοδοι εφαρμοσμένης μηχανικής γνώσης, η βασική προσέγγιση είναι η ίδια: ένας μηχανικός γνώσης παίρνει συνέντευξη και παρατηρεί έναν ανθρώπινο εμπειρογνώμονα ή μια ομάδα εμπειρογνωμόνων και μαθαίνει τι οι εμπειρογνώμονες ξέρουν, και πώς χειρίζονται τη γνώση τους. Ο μηχανικός μεταφράζει έπειτα τη γνώση σε γλώσσα που να μπορεί να χειριστεί ο υπολογιστής, και σχεδιάζει μια μηχανή συμπεράσματος, μια δομή συλλογισμού, η οποία χρησιμοποιεί τη γνώση κατάλληλα. Καθορίζει επίσης πώς να ενσωματώσει τη χρήση της αβέβαιης γνώσης στη διαδικασία συλλογισμού, και ποια είδη εξήγησης θα ήταν χρήσιμα στον τελικό χρήστη.

Έπειτα, η μηχανή συμπεράσματος και οι εγκαταστάσεις για την αναπαράσταση της γνώσης και για τις επεξηγήσεις είναι προγραμματισμένες, και το πεδίο γνώσης εισάγεται στο πρόγραμμα κομμάτι-κομμάτι. Μπορεί η μηχανή συμπεράσματος να μην είναι σωστή, η μορφή αναπαράστασης γνώσης να είναι ακατάλληλη για το είδος γνώσης που απαιτείται για την εργασία, και ο εμπειρογνώμονας να αποφασίσει ότι τα κομμάτια της γνώσης είναι λανθασμένα. Όλα αυτά ανακαλύπτονται και τροποποιούνται καθώς το έμπειρο σύστημα κερδίζει βαθμιαία την ικανότητα και τη γνώση να διακρίνει αυτές τις αδυναμίες.

Η ανακάλυψη και η συσσώρευση των τεχνικών συλλογισμού των μηχανών και της αναπαράστασης γνώσης είναι γενικά το αντικείμενο μελέτης της τεχνητής νοημοσύνης. Η ανακάλυψη και η συσσώρευση της γνώσης ενός πεδίου εργασιών, είναι η περιοχή με την οποία ασχολούνται οι ειδικοί του αντικειμένου. Η εξειδικευμένη γνώση αποτελείται εξίσου από την επίσημη γνώση εγχειριδίων και από εμπειρική γνώση - την πείρα των εμπειρογνωμόνων.



Σχήμα 3. Αναλυτική αρχιτεκτονική ενός έμπειρου συστήματος.

Πιο αναλυτικά, ένα έμπειρο σύστημα αποτελείται βασικά από τέσσερα σημαντικά συστατικά:

1. Βάση γνώσεων- Knowledge Base. Είναι η γνώση στο έμπειρο σύστημα, που κωδικοποιείται σε μια μορφή που το σύστημα μπορεί να χρησιμοποιήσει. Αναπτύσσεται από κάποιο συνδυασμό ανθρώπων (παραδείγματος χάριν, ένας μηχανικός γνώσης) και ενός αυτοματοποιημένου συστήματος εκμάθησης (παραδείγματος χάριν, ένα σύστημα που μπορεί να μάθει μέσω της ανάλυσης των καλών παραδειγμάτων της απόδοσης εμπειρογνωμόνων).

2. Επίλυση προβλήματος- Problem Solver. Είναι ένας συνδυασμός αλγορίθμων και θεωριών σχεδιασμένων με σκοπό να χρησιμοποιήσει τη βάση γνώσεων σε μία προσπάθεια να λυθούν τα προβλήματα σε έναν συγκεκριμένο τομέα.

3. Πληροφοριοδότης- Communicator. Έχει ως σκοπό να διευκολύνει την κατάλληλη επικοινωνία ανάμεσα στους δημιουργούς του έμπειρου

συστήματος και στους χρήστες του.

4. Εξήγηση και βοήθεια. Έχει ως σκοπό να παρέχει τη βοήθεια στο χρήστη και επίσης να παρέχει λεπτομερείς εξηγήσεις για το «τι και γιατί» των δραστηριοτήτων του έμπειρου συστήματος κατά τη διαδικασία επίλυσης του προβλήματος.

Είναι πολύ σημαντικό να γίνει κατανοητή η στενή ειδίκευση του χαρακτηριστικού έμπειρου συστήματος. Ένα έμπειρο σύστημα που έχει χτιστεί με σκοπό να καθορίσει εάν ένα πρόσωπο που υποβάλλει αίτηση για ένα δάνειο είναι ένας αποδεκτός κίνδυνος δανείου, δεν μπορεί να εντοπίσει τις μολυσματικές ασθένειες, και αντίστροφα, ‘όπως και ένα έμπειρο σύστημα με σκοπό να βοηθήσει έναν δικηγόρο να εξετάσει τη νομολογία δεν μπορεί να βοηθήσει έναν καθηγητή λογοτεχνίας να αναλύσει την ποίηση.

Οι ερευνητές της ΤΝ βασίζουν συχνά την εργασία τους σε μια προσεκτική μελέτη του πώς οι άνθρωποι λύνουν τα προβλήματα, και στην ανθρώπινη νοημοσύνη. Στην προσπάθεια να αναπτυχθούν αποτελεσματικά συστήματα ΤΝ, μαθαίνουν για τις ανθρώπινους ικανότητες και τους περιορισμούς της. Ένα από τα ενδιαφέροντα πράγματα που προκύπτει από τη μελέτη των έμπειρων συστημάτων, είναι ότι μέσα σε έναν τομέα στενής ειδίκευσης, ένας ανθρώπινος εμπειρογνώμονας μπορεί να χρησιμοποιεί μόνο μερικές εκατοντάδες ως και μερικές χιλιάδες κανόνες, σε αντίθεση με ένα έμπειρο σύστημα που έχει μεγαλύτερη δυνατότητα επεξεργασίας κανόνων.

2.6 Εργαλεία, κελύφη

Σε σύγκριση με τη μεγάλη ποικιλία που βρίσκουμε στα πεδία γνώσης, μόνο ένας μικρός αριθμός μεθόδων ΤΝ μπορεί να χρησιμοποιηθεί στα έμπειρα συστήματα. Δηλαδή, υπάρχουν αυτή τη στιγμή μόνο ελάχιστοι συγκριτικά τρόποι για την αναπαράσταση γνώσης, ή τη διεξαγωγή συμπερασμάτων, ή την παραγωγή εξηγήσεων. Κατά συνέπεια, μπορούν να χτιστούν συστήματα που περιέχουν αυτές τις χρήσιμες μεθόδους, χωρίς να

περιέχουν καμία συγκεκριμένη γνώση που να αφορά το πεδίο γνώσης. Τέτοια συστήματα είναι γνωστά ως σκελετικά συστήματα (skeletal systems), κελύφη (shells), ή απλά εργαλεία ΤΝ (Artificial Intelligence tools).

Η κατασκευή των έμπειρων συστημάτων με τη χρήση των κελυφών προσφέρει σημαντικά πλεονεκτήματα. Ένα σύστημα μπορεί να χτιστεί για να εκτελέσει έναν μοναδικό στόχο, απλά με την εισαγωγή σε ένα κέλυφος όλης της απαραίτητης γνώσης για ένα πεδίο εργασιών. Η μηχανή συμπεράσματος που εφαρμόζει τη γνώση στο στόχο που πραγματευόμαστε κατασκευάζεται μέσω του κελύφους. Εάν το πρόγραμμα δεν είναι πολύ περίπλοκο και εάν ένας εμπειρογνώμονας έχει κάποια εμπειρία στη χρήση ενός κελύφους, ο εμπειρογνώμονας μπορεί να εισαγάγει τη γνώση ο ίδιος, χωρίς περαιτέρω βοήθεια.

Αν και τα κελύφη απλοποιούν τον προγραμματισμό, όμως γενικά δεν βοηθούν ιδιαίτερα στην απόκτηση της γνώσης. Η απόκτηση γνώσης έχει περισσότερο να κάνει με την παροχή γνώσης στα έμπειρα συστήματα, ένας στόχος που εκτελείται ακόμα ως και σήμερα από τους μηχανικούς γνώσης. Η επιλογή της μεθόδου συλλογισμού, ή του κελύφους, είναι σημαντικές, αλλά αυτό δεν τόσο σημαντικό όσο η απόκτηση υψηλής ποιότητας γνώσης. Η δύναμη ενός έμπειρου συστήματος έγκειται στη συσσώρευση γνώσης για το πεδίο εργασιών- όσο περισσότερη γνώση αποκτά ένα σύστημα, τόσο πιο ικανό γίνεται.

2.7 Χαρακτηριστικά Έμπειρων Συστημάτων έναντι των Συμβατικών Προγραμμάτων

Τα χαρακτηριστικά των έμπειρων συστημάτων έναντι αυτών των συμβατικών προγραμμάτων συνοψίζονται στον Πίνακα 1.

Έμπειρα Συστήματα	***Συμβατικά Προγράμματα***
Προσομοιώνουν τον τρόπο επίλυσης ενός προβλήματος	Προσομοιώνουν το ίδιο το πρόβλημα
Παράσταση και χειρισμός γνώσης σε επίπεδο συμβόλων	Παράσταση και χειρισμός δεδομένων σε επίπεδο αριθμητικών υπολογισμών
Χρήση ευρετικών μεθόδων για περιορισμό του χώρου αναζήτησης	Χρήση αλγορίθμων
Χρήση γλωσσών που πλησιάζουν την ανθρώπινη	Χρήση γλωσσών που βρίσκονται πλησιέστερα στον τρόπο λειτουργίας του Η/Υ
Βάση γνώσης (δεδομένα και εξαγωγή συμπερασμάτων)	Βάση δεδομένων - η γνώση ενσωματώνεται στο πρόγραμμα
Ευχέρεια στην επέκταση και αναθεώρηση της γνώσης	Η αναθεώρηση της γνώσης επιβάλλει ευρείας κλίμακας μεταβολές στο πρόγραμμα
Χειρισμός ασαφούς, αβέβαιης και μη-πλήρους γνώσης	Δυσχέρεια στο χειρισμό ασαφούς, αβέβαιης και μη-πλήρους γνώσης
Δυνατότητα μη μονότονης συλλογιστικής Δυσχέρεια στη χρήση μη μονότονης λογικής	Επεξήγηση του δρόμου συλλογισμού Ανυπαρξία επεξήγησης

Πίνακας 1. Χαρακτηριστικά Έμπειρων Συστημάτων - Συμβατικών Προγραμμάτων

2.8 Πλεονεκτήματα- Μειονεκτήματα ΕΣ σε σχέση με τον Άνθρωπο-Ειδικό

Στους πίνακες 2 και 3 φαίνονται τα πλεονεκτήματα και μειονεκτήματα των έμπειρων συστημάτων.

Πλεονεκτήματα Ειδικού	***Μειονεκτήματα ΕΣ***
Δημιουργικότητα, Ευρύννοια	Απουσία έμπνευσης, Περιορισμένο πεδίο σκέψης
Κοινή λογική	Δυσχέρεια στη μεταφύτευση της κοινής λογικής
Γνώση των ορίων και δυνατοτήτων τους (μετά-γνώση)	Έλλειψη μετά-γνώσης
Εκφραστική και λειτουργική επεξήγηση του τρόπου σκέψης τους	Μηχανική επεξήγηση του τρόπου λήψης απόφασης
Ο έλεγχος της γνώσης γίνεται υποσυνείδητα	Πρέπει η γνώση να ελέγχεται για ορθότητα, πληρότητα και συνέπεια
Αυτονομία στη μάθηση	Πρέπει να προγραμματιστούν για να μαθαίνουν αυτόματα
Απόκριση σε πραγματικό χρόνο	Δυσκολία απόκρισης σε πραγματικό χρόνο

Πίνακας 2. Μειονεκτήματα ΕΣ- Πλεονεκτήματα ειδικού

Πλεονεκτήματα ΕΣ	***Μειονεκτήματα Ειδικού***
Γνώση πάντα διαθέσιμη	Γνώση διαθέσιμη όταν ο ίδιος είναι παρών
Ευκολία μεταφοράς-αποτύπωσης γνώσης	Δυσκολία μεταφοράς-αποτύπωσης γνώσης
Εργάζεται με συνέπεια	Συναισθηματικές παρορμήσεις
Εργάζεται οπουδήποτε	Η απόδοσή του επηρεάζεται από εξωγενείς παράγοντες
Χαμηλό κόστος λειτουργίας / Υψηλό κόστος ανάπτυξης	Υψηλό κόστος
Αντικειμενικότητα αν η γνώση προέρχεται από πολλούς Ειδικούς	Υποκειμενικότητα

Πίνακας 3. Πλεονεκτήματα ΕΣ- Μειονεκτήματα ειδικού

2.9 Εξέλιξη Γλωσσών Προγραμματισμού της ΤΝ

Η θεμελιώδης υπόθεση εργασίας της ΤΝ είναι ότι η ευφυής συμπεριφορά μπορεί να περιγραφεί ακριβώς με το χειρισμό συμβόλων και μπορεί να διαμορφωθεί με τις ικανότητες επεξεργασίας συμβόλων του υπολογιστή.

Προς το τέλος της δεκαετίας του '50, εφευρέθηκαν πρόσθετες γλώσσες προγραμματισμού που διευκολύνουν το χειρισμό συμβόλων. Η πιο γνωστή καλείται LISP (LISt Processing- επεξεργασία καταλόγων). Λόγω της απλότητας και της ευελιξίας της, τα περισσότερα ερευνητικά προγράμματα ΤΝ γράφονται στη LISP, αλλά οι εμπορικές εφαρμογές δεν τη χρησιμοποιούν και τόσο.

Στις αρχές της δεκαετίας του 1970 μια άλλη γλώσσα προγραμματισμού ΤΝ εφευρέθηκε στη Γαλλία. Καλείται PROLOG (PROgramming in LOGic- προγραμματισμός με λογική). Τα προγράμματα που έχουν γραφτεί σε PROLOG έχουν παρόμοια συμπεριφορά με συστήματα γραμμένα σε LISP. Η PROLOG όμως, δεν κέρδισε απευθείας την υποστήριξη των προγραμματιστών για την ΤΝ. Στις αρχές της δεκαετίας του 1980 ανακοινώθηκε από τους Ιάπωνες ότι θα χρησιμοποιούσαν μια γλώσσα προγραμματισμού λογικής για το Πέμπτο Πρόγραμμα Συστημάτων Υπολογισμού Παραγωγής (Fifth Generation Computing Systems - FGCS). Από τότε έχουν προκύψει ποικίλες γλώσσες προγραμματισμού βασισμένες στη λογική, και ο όρος Prolog έχει γίνει γενικός αποδεκτός.

Παρακάτω, παραθέτουμε μερικά από τα πιο γνωστά ΕΣ με τα κυριότερα χαρακτηριστικά τους:

- DENDRAL

  Ταυτοποίηση χημικών ενώσεων μέσω φασματικής ανάλυσης.

  Χρήση ευρετικών κανόνων για περιορισμό του χώρου αναζήτησης.

- MYCIN

  Διάγνωση και θεραπεία της μηνιγγίτιδας και της βακτηριαιμίας.

Χρήση συντελεστή βεβαιότητας για τις λύσεις, λόγω αβεβαιότητας απαντήσεων χρήστη.

- PROSPECTOR

  Πρόβλεψη της ακριβούς θέσης ορυκτών κοιτασμάτων αξιοποιώντας γεωλογικά δεδομένα.

  Χρήση σημασιολογικών δικτύων και δικτύων πιθανοτήτων.

- INTERNIST

  Διάγνωση παθολογικών περιπτώσεων με πολύ μεγάλο αριθμό εναλλακτικών διαγνώσεων.

  Χρήση ευρετικής συλλογιστικής (απαγωγική) για την πιθανότερη διάγνωση.

- XCON

  Διαμόρφωση υπολογιστών DEC, για να ανταποκρίνονται στις προδιαγραφές του πελάτη.

  Αναζήτηση κατάλληλου συνδυασμού και χωρικής διάταξη των εξαρτημάτων, με αποφυγή των ασυμβατοτήτων λειτουργίας και διασύνδεσης μεταξύ τους.

Στη συνέχεια, θα ασχοληθούμε κατά κύριο λόγο με δυο έμπειρα συστήματα που αναφέρθηκαν νωρίτερα, κάνοντας πρακτική εφαρμογή τόσο στη Fuzzy Logic αλλά και στους συντελεστές βεβαιότητας. Τα δύο προγράμματα αυτά είναι το FuzzyCLIPS και μια τροποποίηση του MYCIN, αντίστοιχα, γι’ αυτό και παρακάτω αναλύονται περισσότερο.

3. Το πρόγραμμα CLIPS

3.1 Γενικά

Το πρόγραμμα CLIPS (C Language Integrated Production System) είναι ένα παραγωγικό εργαλείο έμπειρων συστημάτων ανάπτυξης και παράδοσης που παρέχει ένα πλήρες περιβάλλον για την κατασκευή έμπειρων συστημάτων βασισμένων σε κανόνες ή/ και σε αντικείμενα (rule and/ or object based). Δημιουργήθηκε το 1985 στο Κέντρο Ερευνών Johnson Space της NASA, και χρησιμοποιείται τώρα ευρέως στην κυβέρνηση, τη βιομηχανία, και τον ακαδημαϊκό κόσμο.

Τα κύρια χαρακτηριστικά του είναι:

- Αντιπροσώπευση γνώσης (Knowledge Representation): Το CLIPS παρέχει ένα πλήρες εργαλείο για μια ευρεία ποικιλία γνώσης με την υποστήριξη για τρία διαφορετικά παραδείγματα προγραμματισμού: βασισμένος στους κανόνες (rule-based), αντικειμενοστραφής (object-oriented) και διαδικαστικός (procedural).

  Ο βασισμένος στους κανόνες προγραμματισμός επιτρέπει στη γνώση για να αντιπροσωπευθεί ως ευρετική, η όποια καθορίζει ένα σύνολο ενεργειών που εκτελούνται για μια δεδομένη κατάσταση.

  Ο αντικειμενοστραφής προγραμματισμός επιτρέπει στα πολυσύνθετα συστήματα να παίρνουν μορφή με πιο απλά συστατικά (έτσι ώστε να μπορούν να επαναχρησιμοποιηθούν εύκολα για να διαμορφώσουν άλλα συστήματα ή για να δημιουργήσουν καινούρια συστατικά).

  Οι ικανότητες του διαδικαστικού προγραμματισμού που παρέχονται από το CLIPS είναι παρόμοιες με τις ικανότητες που βρίσκονται σε γλώσσες όπως η C, η Java, η Ada, και η LISP.

- Φορητότητα (Portability): Το CLIPS είναι γραμμένο σε C για φορητότητα και ταχύτητα, και έχει εγκατασταθεί σε πολλά διαφορετικά λειτουργικά συστήματα χωρίς να χρειαστούν αλλαγές κώδικα. Τα λειτουργικά συστήματα στα οποία έχει εξεταστεί το CLIPS περιλαμβάνουν τα Windows XP, MacOS X, και Unix. Το

CLIPS μπορεί να εφαρμοστεί σε οποιοδήποτε σύστημα έχει C ή C++ compiler. Το CLIPS παρέχεται με όλους τους κωδικούς που μπορεί να απαιτούνται για να τροποποιηθεί ή να προσαρμοστεί για να καλύψει τις συγκεκριμένες ανάγκες ενός χρήστη.

- Ολοκλήρωση/ Έκταση (Integration/ Extensibility): Το CLIPS μπορεί να ενσωματωθεί στον διαδικαστικό κώδικα, να κληθεί ως υπορουτίνα, και να λειτουργήσει με γλώσσες όπως η C, η Java, η FORTRAN και η ADA. Το CLIPS μπορεί να επεκταθεί εύκολα από τον χρήστη, μέσω της χρήσης διαφόρων σαφέστατα καθορισμένων πρωτοκόλλων.

- Διαλογική ανάπτυξη (Interactive Development): Η στάνταρ έκδοση του CLIPS παρέχει ένα διαλογικό περιβάλλον ανάπτυξης (οι εντολές δίδονται μέσω κειμένου), συμπεριλαμβανομένων των βοηθειών διόρθωσης, on-line βοήθεια, και ένας ενσωματωμένος συντάκτης (ClipsEdit). Ακόμα, έχουν αναπτυχθεί εφαρμογές που ευνοούν τη διεπαφή με τον χρήστη, όπως κυλιόμενο μενού επιλογών, ενσωματωμένοι συντάκτες, και πολλαπλά παράθυρα για το MacOS, τα Windows XP, και το περιβάλλον των Windows X.

- Επαλήθευση/ Επικύρωση (Verification/ Validation): Το CLIPS περιλαμβάνει διάφορα χαρακτηριστικά γνωρίσματα για να υποστηρίξει την επαλήθευση και την επικύρωση των έμπειρων συστημάτων, συμπεριλαμβανομένης της υποστήριξης για τον μοριακό σχεδιασμό και τον διαχωρισμό μιας βάσης γνώσεων, τον στατικό και δυναμικό περιορισμό των ελέγχων των τιμών και των συναρτήσεων επιχειρημάτων, και μια σημασιολογική ανάλυση των σχημάτων κανόνων για να καθορίσουν εάν οι ασυνέπειες θα μπορούσαν να αποτρέψουν έναν κανόνα να εκτελεστεί ή να παραγάγει ένα λάθος.

- Πλήρης τεκμηρίωση (Fully Documented): Το CLIPS παρέχεται με αναλυτικές οδηγίες, οι οποίες συμπεριλαμβάνονται σε ένα εγχειρίδιο αναφοράς και έναν οδηγό χρήστη [2].

- Χαμηλότερο κόστος: Το CLIPS αποτελεί ακόμα και σήμερα ένα λογισμικό ελεύθερα προσβάσιμο στο ευρύ κοινό [4].

3.2 Ασαφής Λογική και Συντελεστές Βαρών

Αν και ένα έμπειρο σύστημα αποτελείται πρώτιστα από μια βάση γνώσεων και μια μηχανή συμπεράσματος, μερικά άλλα χαρακτηριστικά γνωρίσματα αξίζουν να αναφερθούν: συλλογισμός με χρήση αβεβαιότητας, και αιτιολόγηση της γραμμής συλλογισμού.

Η γνώση είναι σχεδόν πάντα ελλιπής και αβέβαιη. Για να εξετάσουμε την αβέβαιη γνώση, ένας κανόνας μπορεί να είχε συνδέσει με αυτή έναν παράγοντα εμπιστοσύνης ή ένα βάρος. Το σύνολο μεθόδων για την αβέβαιη γνώση σε συνδυασμό με τα αβέβαια δεδομένα στη διαδικασία συλλογισμού καλείται συλλογισμός με αβεβαιότητα. Μια σημαντική υποκατηγορία των μεθόδων για τον συλλογισμό με αβεβαιότητα καλείται "fuzzy logic" (ασαφής λογτική) και τα συστήματα που τους χρησιμοποιούν είναι γνωστά ως "fuzzy systems" (ασαφή συστήματα).

Αναλυτικά για το πώς χρησιμοποιούνται τα βάρη για την αντιμετώπιση της αβέβαιης γνώσης θα μιλήσουμε σε επόμενο κεφάλαιο.

3.3 Ασαφής Λογική - FuzzyCLIPS

Η ασαφής λογική (Fuzzy Logic) είναι μια μορφή λογικής πολλών μεταβλητών που προέρχεται από τη συγκεχυμένη καθορισμένη θεωρία για να εξετάσει το συλλογισμό που είναι περισσότερο κατά προσέγγιση, παρά ακριβής. Σε αντίθεση με τα δυαδικά σύνολα που έχουν τη δυαδική λογική, επίσης γνωστή ως crisp logic, οι μεταβλητές συγκεχυμένης λογικής μπορούν να παίρνουν τιμές και διάφορες του 0 ή του 1. Ο βαθμός αλήθειας μιας δήλωσης μπορεί να κυμανθεί μεταξύ 0 και 1 και δεν περιορίζεται στις δύο τιμές αλήθειας {αληθές (1), ψευδές (0)} όπως στην κλασική προτασιακή λογική. Όταν χρησιμοποιούνται οι γλωσσικές μεταβλητές, αυτοί οι βαθμοί μπορούν να ρυθμιστούν από τις συγκεκριμένες λειτουργίες, ανάλογα με τη βεβαιότητα εκπλήρωσης ενός γεγονότος, ή όχι.

Ο όρος " Fuzzy logic" αποτελεί συνεπεία της ανάπτυξης της θεωρίας των ασαφών συνόλων (fuzzy sets) από τον Lotfi Zadeh. Το 1965, ο Lotfi Zadeh πρότεινε τη θεωρία ασαφών συνόλων (fuzzy set theory), και καθιέρωσε αργότερα την ασαφή λογική βασισμένη στα ασαφή σύνολα. Η ασαφής λογική έχει εφαρμοστεί σε διάφορους τομείς, από τη θεωρία ελέγχου ως την τεχνητή νοημοσύνη, όμως ακόμα παραμένει αμφισβητούμενη μεταξύ των περισσοτέρων μελετητών της Στατιστικής, που προτιμούν τη Μπεϋζιανή λογική (Bayesian logic), και μερικών μηχανικών ελέγχου, οι οποίοι προτιμούν την παραδοσιακή λογική των δυο μεταβλητών (two-valued logic).

Το FuzzyCLIPS [1] είναι μια επέκταση του κελύφους ασαφούς λογικής του έμπειρου συστήματος CLIPS από τη NASA. Αναπτύχθηκε από την Integrated Reasoning Group του Ιδρύματος για την Τεχνολογία Πληροφοριών του Εθνικού Συμβουλίου Έρευνας του Καναδά (Institute for Information Technology of the National Research Council of Canada) και έχει διανεμηθεί ελεύθερα στο κοινό εδώ και πολλά χρόνια. Ενισχύει το έμπειρο σύστημα CLIPS με την παροχή μιας ικανότητας συλλογισμού με αβεβαιότητα που είναι πλήρως ενσωματωμένη στη μηχανή γεγονότων και συμπεράσματος CLIPS, επιτρέποντας στον χρήστη να αντιπροσωπεύσει και να χειριστεί τα αβέβαια γεγονότα και τους κανόνες. Το FuzzyCLIPS μπορεί να εξετάσει τον ακριβή, ασαφή (ή ανακριβή), και συνδυασμένο συλλογισμό, επιτρέποντας στους ασαφής και κανονικούς όρους για να αναμιχθούν ελεύθερα στους κανόνες και τα γεγονότα ενός έμπειρου συστήματος. Το σύστημα χρησιμοποιεί τις δύο βασικές ανακριβείς έννοιες, ασάφεια και αβεβαιότητα. Παρέχει ένα χρήσιμο περιβάλλον για την ανάπτυξη των ασαφών εφαρμογών, αλλά απαιτεί σημαντική προσπάθεια για να ενημερώνεται και να κυριαρχεί στο χώρο, καθώς συνεχώς κυκλοφορούν νέες εκδόσεις του CLIPS [4].

4. Έμπειρο σύστημα MYCIN

4.1 Γενικά

Το MYCIN είναι ένα έμπειρο σύστημα που αναπτύχθηκε στο Πανεπιστήμιο του Στάνφορντ στη δεκαετία του '70. Σκοπός του ήταν να εντοπίσει και να συστήσει την θεραπευτική αγωγή που θα ακολουθηθεί σε ορισμένες μολύνσεις αίματος. Για να γίνει μια διάγνωση σωστά, απαιτείται να παραχθούν καλλιέργειες του μολυσμένου οργανισμού. Δυστυχώς, αυτό απαιτεί περίπου 48 ώρες, και εάν οι γιατροί περίμεναν έως να ολοκληρωθούν, ο ασθενής τους θα μπορούσε να είναι νεκρός! Έτσι, οι γιατροί πρέπει να προβούν σε γρήγορες εικασίες για τα πιθανά προβλήματα από τα διαθέσιμα στοιχεία που έχουν στα χέρια τους, και να χρησιμοποιήσουν αυτές τις εικασίες για να παρέχουν αγωγή που να καλύπτει τις πιθανές μολύνσεις, όπου δίδονται φάρμακα που θα αντιμετωπίσουν οποιοδήποτε πιθανό πρόβλημα.

Το MYCIN αναπτύχθηκε εν μέρει προκειμένου να εξερευνηθεί το πώς οι ανθρώπινοι εμπειρογνώμονες λαμβάνουν αυτές τις 'κατά προσέγγιση' (αλλά σημαντικές) εικασίες βασισμένοι στις μερικές πληροφορίες. Εντούτοις, το πρόβλημα είναι επίσης σημαντικό σε πρακτική βάση, καθώς υπάρχουν πολλοί κατώτεροι ή μη εξειδικευμένοι γιατροί, που πρέπει μερικές φορές να κάνουν μια τέτοια 'κατά προσέγγιση' διάγνωση, και εάν υπάρχει ένα ειδικό εργαλείο διαθέσιμο για να απευθυνθούν, μπορεί να βοηθήσει να δοθεί πιο κατάλληλη θεραπεία. Στην πραγματικότητα, το MYCIN δεν χρησιμοποιήθηκε ποτέ πραγματικά στην πράξη. Κι αυτό όχι λόγω οποιασδήποτε αδυναμίας στην απόδοσή του (στις δοκιμές ξεπέρασε τα μέλη της Ιατρικής Σχολής του Στάνφορντ). Ήταν τόσο λόγω ηθικής, αλλά και νομικών θεμάτων σχετικών με τη χρήση των υπολογιστών στην ιατρική: εάν δώσει λανθασμένη διάγνωση, ποιός πρέπει να μηνυθεί;

Το MYCIN αντιπροσωπεύει τη γνώση του ως σύνολο IF-THEN κανόνων με συντελεστές βεβαιότητας. Τα παρακάτω είναι μια αγγλική εκδοχή ενός από τους κανόνες του MYCIN:

IF the infection is pimary- bacteremia

AND the site of the culture is one of the sterile sites

AND the suspected portal of entry is the gastrointestinal tract

THEN there is suggestive evidence (0.7) that infection is bacteroid.

Το 0.7 είναι κατά προσέγγιση η βεβαιότητα ότι το συμπέρασμα θα είναι αληθινό λαμβάνοντας υπόψη τα στοιχεία. Εάν τα στοιχεία είναι αβέβαια, οι βεβαιότητες των κομματιών των στοιχείων θα συνδυαστούν με τη βεβαιότητα του κανόνα για να δοθεί η βεβαιότητα του συμπεράσματος.

Το MYCIN γράφτηκε σε LISP, και οι κανόνες του αντιπροσωπεύονται τυπικά ως εκφράσεις της LISP. Το μέρος δράσης του κανόνα θα μπορούσε ακριβώς να είναι ένα συμπέρασμα για το πρόβλημα που λύνεται, ή θα μπορούσε να είναι μια αυθαίρετη έκφραση της LISP. Αυτό παρέχει μεγάλη ευελιξία, αλλά εάν εξαιρεθεί η απλότητα και η σαφήνεια των βασισμένων στους κανόνες συστημάτων, η χρήση του προγράμματος θα έπρεπε να γίνει με μεγάλη προσοχή.

Εν πάση περιπτώσει, το MYCIN είναι (πρώτιστα) ένα κατευθυνμένο στο στόχο (goal-directed) σύστημα, που χρησιμοποιεί τη βασική οπίσθια αλυσιδωτή (backward chaining) στρατηγική συλλογισμού που περιγράψαμε παραπάνω. Εντούτοις, το MYCIN χρησιμοποίησε διάφορες πιθανές ευρετικές εκδοχές (heuristics) για να ελέγξει την αναζήτηση μιας λύσης (ή την απόδειξη κάποιας υπόθεσης). Αυτές απαιτήθηκαν και για να καταστήσουν το συλλογισμό αποδοτικό, αλλά και για να αποτρέψουν το χρήστη να πρέπει να απαντήσει σε πάρα πολλές περιττές ερωτήσεις.

Μια στρατηγική πρέπει να υποβάλλει αρχικά το χρήστη σε διάφορες λίγο- πολύ προετοιμασμένες (στάνταρ) ερωτήσεις, οι οποίες πάντα απαιτούνται, και που επιτρέπουν στο σύστημα να αποκλείσουν τις εντελώς απίθανες διαγνώσεις. Μόλις υποβληθούν αυτές οι ερωτήσεις, το σύστημα μπορεί έπειτα να εστιάσει στις ιδιαίτερες, πιο συγκεκριμένες πιθανές παθήσεις του αίματος, και να χρησιμοποιήσει τη βασική οπίσθια αλυσιδωτή στρατηγική για να προσπαθήσει να αποδείξει την κάθε μια από αυτές. Αυτό αποκλείει πολλή περιττή αναζήτηση, και ακολουθεί επίσης το μοτίβο των 'συνεντεύξεων' ασθενή- ιατρού.

Οι άλλες στρατηγικές αφορούν τον τρόπο με τον οποίο οι κανόνες καλούνται να εκτελεστούν. Η πρώτη είναι απλή: λαμβάνοντας υπόψη έναν πιθανό κανόνα για να χρησιμοποιήσει, το MYCIN πρώτα ελέγχει όλες τις πτυχές του κανόνα, για να δει εάν οποιεσδήποτε είναι γνωστό ότι είναι

αναληθείς. Εάν ισχύει κάτι τέτοιο, δεν υπάρχει κανένα νόημα να γίνει χρήση του κανόνα. Οι άλλες στρατηγικές αφορούν περισσότερους τους συντελεστές βεβαιότητας. Το MYCIN θα εξετάσει αρχικά τους κανόνες που έχουν τα πιο βέβαια συμπεράσματα, και θα εγκαταλείψει μια αναζήτηση μόλις οι βεβαιότητες που εμφανίζονται πάρουν τιμές κάτω από 0.2.

Υπήρξαν πολλές άλλες εξελίξεις από το πρόγραμμα MYCIN. Παραδείγματος χάριν, το EMYCIN είναι πραγματικά το πρώτο ειδικό κέλυφος που αναπτύχθηκε από το MYCIN. Ένα νέο έμπειρο σύστημα αποκαλούμενο PUFF αναπτύχθηκε χρησιμοποιώντας το EMYCIN στη νέα περιοχή των καρδιακών παθήσεων. Και το σύστημα NEOMYCIN αναπτύχθηκε για την κατάρτιση των γιατρών, και το οποίο θα τους εφάρμοζε ποικίλες περιπτώσεις παραδειγμάτων, ελέγχοντας τα συμπεράσματά τους και εξηγώντας τους σε ποιά σημεία έκαναν λάθος [5].

4.2 Συντελεστές Βαρών

Το έμπειρο σύστημα MYCIN ήταν και το πρώτο έμπειρο σύστημα που χρησιμοποίησε τους συντελεστές βεβαιότητας. Έτσι εισήγαγε τον τύπο

$$CF = CF1 + CF2 - CF1*CF2$$

για το συνδυασμό δυο προβλέψεων με το ίδιο συμπέρασμα στην περίπτωση που έχουμε θετικούς συντελεστές.

Ενώ όμως ο τύπος αυτός λειτουργεί σε ορισμένα προβλήματα δίδοντας επαρκώς αποδεκτά αποτελέσματα, στην περίπτωση που ο ένας από τους δυο κανόνες παράγει πιο αξιόπιστες προβλέψεις, το μοντέλο του MYCIN δεν παράγει ικανοποιητικά αποτελέσματα.

Οδηγούμαστε λοιπόν, στην αναζήτηση ενός πιο αποτελεσματικού τύπου, όπως ο πιο γενικευμένος τύπος υπολογισμού τελικής πρόβλεψης που χρησιμοποιήθηκε στο έμπειρο σύστημα PASS (Hatzilygeroudis, Karatrantou και Pierrakeas, 2004):

$$CF = w1 * CF1 + w2 * CF2 + w * CF1*CF2,$$

όπου τα w1, w2, w είναι αριθμητικά βάρη που ικανοποιούν τη σχέση

$$w1 + w2 + w = 1, \text{ ώστε } 0 \leq CF \leq 1.$$

Παρατηρούμε ότι ο τύπος του MYCIN συμπεριλαμβάνεται στον παραπάνω γενικευμένο τύπο, αφού προκύπτει από αυτόν εάν θέσουμε στα βάρη τιμές w1 = w2 = 1 και w = -1.

Με την εισαγωγή των συντελεστών βαρών στον τύπο συνδυασμού προβλέψεων, λύνεται το πρόβλημα που εκθέσαμε νωρίτερα, αφού μπορούμε να αποδώσουμε μεγαλύτερη τιμή στο βάρος της πιο αξιόπιστης πρόβλεψης, και να έχουμε πιο ικανοποιητικά αποτελέσματα.

4.3 Σύστημα Υλοποίησης και Αξιολόγησης Μεθόδου Μετάδοσης Συντελεστών Βεβαιότητας

Το πρόγραμμα αυτό έχει ως σκοπό μέσω της ίδιας εφαρμογής και τη χρήση ενός dataset, να συγκρίνει τη μέθοδο του MYCIN και τη γενικευμένη μέθοδο με χρήση βαρών. Λειτουργεί ως εξής: από το ίδιο dataset από στιγμιότυπα ενός προβλήματος δημιουργείται έμπειρο σύστημα, το οποίο παράγει δύο προβλέψεις με συντελεστές βεβαιότητας. Οι προβλέψεις αυτές, συνδυάζονται στη συνέχεια με τους δυο παραπάνω τρόπους ώστε να συγκριθούν.

Κατά τη διαδικασία παραγωγής των προβλέψεων, κάνοντας χρήση των πληροφοριών του προβλήματος και τα στιγμιότυπα του dataset, υπολογίζονται με τη χρήση γενετικού αλγορίθμου τα βέλτιστα βάρη, και το πρόγραμμα εξάγει ένα έμπειρο σύστημα σε γλώσσα CLIPS, το οποίο χρησιμοποιεί τα εξαγόμενα βάρη για να συνδυάσει τις δύο προβλέψεις.

Κατασκευάστηκε στα πλαίσια της Διπλωματικής εργασίας [6]. Για την κατασκευή του συστήματος χρησιμοποιήθηκε το περιβάλλον υλοποίησης DevC++, η βιβλιοθήκη clips.h, και για την παραγωγή του

γενετικού αλγορίθμου έγινε χρήση της βιβλιοθήκης GAlib, ενώ το περιβάλλον διεπαφής υλοποιήθηκε σε Visual C++.

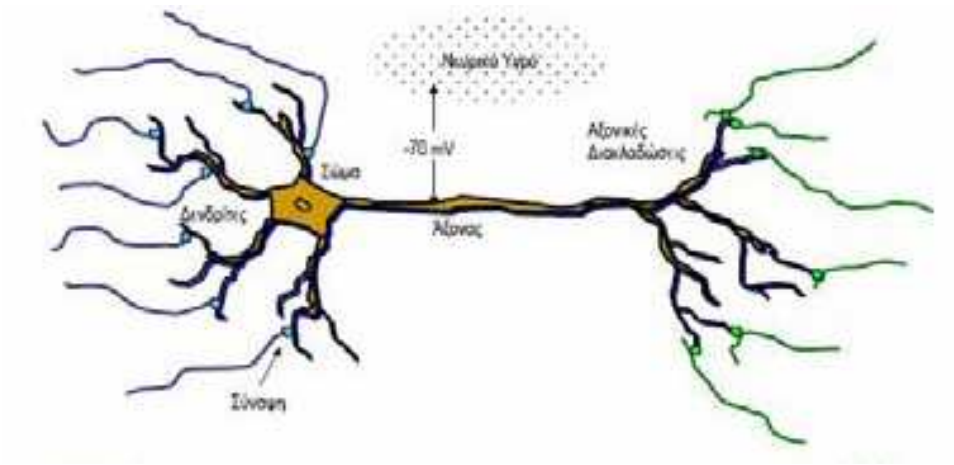
5. Νευρωνικά Δίκτυα

5.1 Γενικά

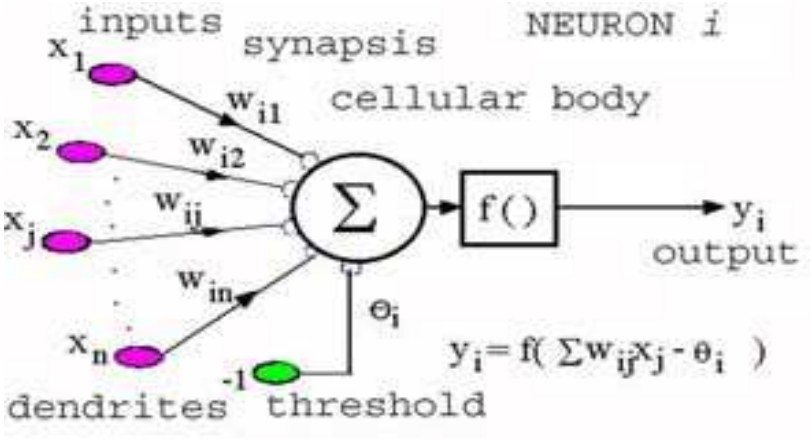
Τα τεχνητά νευρωνικά δίκτυα είναι εμπνευσμένα από τη βιολογία. Αποτελούνται από στοιχεία (τεχνητοί νευρώνες) τα οποία συμπεριφέρονται κατά τρόπο ανάλογο των πιο στοιχειωδών λειτουργιών των βιολογικών κυττάρων. Οι τεχνητοί νευρώνες είναι οργανωμένοι κατά τέτοιο τρόπο, ώστε να προσομοιώνουν την ανατομία του ανθρωπίνου εγκεφάλου. Παρά την, όχι όμως και τόσο μεγάλη ομοιότητα τους με τον εγκέφαλο, επιτυγχάνουν να προσεγγίσουν ένα μεγάλο αριθμό χαρακτηριστικών της δομής αλλά και λειτουργίας του εγκεφάλου. Για παράδειγμα, μαθαίνουν χρησιμοποιώντας εμπειρία την οποία έχουν συσσωρεύσει, έχουν την ικανότητα γενίκευσης από προηγούμενα παραδείγματα σε νέα, μπορούν να επεξεργαστούν μια ομάδα δεδομένων και να ξεχωρίσουν από αυτή τα πιο ουσιώδη χαρακτηριστικά [5].

Η χρησιμοποίηση τεχνικών τεχνητής νοημοσύνης όπως είναι τα νευρωνικά δίκτυα, έχει αρχίσει να βρίσκει εφαρμογή σε προβλήματα Υπολογιστικής Στατικής και Δυναμικής των κατασκευών την τελευταία δεκαετία. Τέτοιου είδους εφαρμογές είναι η ανάλυση αξιοπιστίας κατασκευών, όπου στόχος είναι η πρόβλεψη των αποτελεσμάτων της ανάλυσης, ή προβλήματα βέλτιστου σχεδιασμού, όπου οι απαιτούμενες τιμές των περιορισμών για κάθε νέο σχεδιασμό προέρχονται από την πρόβλεψη ενός κατάλληλα εκπαιδευμένου νευρωνικού δικτύου. Έχουν επίσης εφαρμοστεί σε προβλήματα θραυστομηχανικής.

Ένα σωστά εκπαιδευμένο νευρωνικό δίκτυο μπορεί να παράγει αποδεκτά, από πλευράς ακρίβειας, αποτελέσματα σε σύντομο υπολογιστικό χρόνο. Η ιδιότητα αυτή των Νευρωνικών Δικτύων αποτελεί και το βασικό τους πλεονέκτημα. Η προσέγγιση της λύσης μέσω ενός νευρωνικού δικτύου έχει μεγάλη αξία σε περιπτώσεις χρονοβόρων αναλύσεων όπου είναι αναγκαία μια γρήγορη εκτίμηση της συμπεριφοράς των φορέων [7], [8].



Σχήμα 3. Φυσικός νευρώνας.



Σχήμα 4. Τεχνητός νευρώνας.

5.2 Ορισμός και δομή του νευρωνικού δικτύου

Το νευρωνικό δίκτυο είναι ένα δίκτυο από υπολογιστικούς κόμβους (νευρώνες, νευρώνια), συνδεδεμένους μεταξύ τους. Είναι εμπνευσμένο από το Κεντρικό Νευρικό Σύστημα, το οποίο προσπαθούν να προσομοιώσουν.

Οι νευρώνες είναι τα δομικά στοιχεία του δικτύου. Υπάρχουν δύο είδη νευρώνων, οι νευρώνες εισόδου και οι υπολογιστικοί νευρώνες. Οι νευρώνες εισόδου δεν υπολογίζουν τίποτα, μεσολαβούν ανάμεσα στις εισόδους του δικτύου και τους υπολογιστικούς νευρώνες. Οι υπολογιστικοί νευρώνες πολλαπλασιάζουν τις εισόδους τους με τα συνοπτικά βάρη και υπολογίζουν το άθροισμα του γινομένου. Το άθροισμα που προκύπτει είναι το όρισμα της συνάρτησης ενεργοποίησης.

Εάν x_{ki} είναι η i-οστή είσοδος του k νευρώνα, w_{ki}: το i-οστό συνοπτικό βάρος του k νευρώνα και Φ(-) η συνάρτηση ενεργοποίησης του νευρωνικού δικτύου, τότε η έξοδος y_k του k νευρώνα δίνεται από την εξίσωση:

$$y_k = \Phi (\Sigma i=0,N \quad x_{ki} * w_{ki})$$

Στον k-οστό νευρώνα υπάρχει ένα συνοπτικό βάρος w_{k0} με ιδιαίτερη σημασία, το οποίο καλείται πόλωση ή κατώφλι (bias, threshold). Η τιμή της εισόδου του είναι πάντα η μονάδα, $x_{k0} = 1$. Εάν το συνολικό άθροισμα από τις υπόλοιπες εισόδους του νευρώνα είναι μεγαλύτερο από την τιμή αυτή, τότε ο νευρώνας ενεργοποιείται. Εάν είναι μικρότερο, τότε ο νευρώνας παραμένει ανενεργός. Η ιδέα προέκυψε από τα βιολογικά νευρικά κύτταρα.

5.3 Συναρτήσεις ενεργοποίησης

Η συνάρτηση ενεργοποίησης μπορεί να είναι βηματική (step transfer function), γραμμική (linear transfer function), μη γραμμική (non-linear transfer function), στοχαστική (stochastic transfer function).

Βηματική συνάρτηση ενεργοποίησης

Η βηματική συνάρτηση ενεργοποίησης μπορεί να είναι:

$$\Phi(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

ή οποιαδήποτε άλλη βηματική συνάρτηση.

Γραμμική συνάρτηση ενεργοποίησης

Η γραμμική συνάρτηση ενεργοποίησης μπορεί να είναι:

$$\Phi(x) = x$$

ή οποιαδήποτε άλλη γραμμική συνάρτηση.

Μη γραμμική συνάρτηση ενεργοποίησης

Η μη γραμμική συνάρτηση ενεργοποίησης που χρησιμοποιείται συνήθως στα νευρωνικά δίκτυα καλείται σιγμοειδής συνάρτηση:

$$\Phi(x) = 1 / (1 + e^{x})$$

Εν κατακλείδι, ο εγκέφαλος περιέχει δισεκατομμύρια νευρώνων που διασυνδέουν για να μοιραστούν, να αποθηκεύσουν και να επεξεργαστούν τις πληροφορίες. Όσο περισσότερες συνδέσεις γίνονται από τους νευρώνες, τόσο καλύτερη η μνήμη κάποιου συγκεκριμένου γεγονότος. Κατά τον ίδιο τρόπο, τα τεχνητά νευρωνικά δίκτυα επιδιώκουν να αναδημιουργήσουν τον τρόπο που ο εγκέφαλος λειτουργεί, με το να προσπαθεί να επιτύχει διασυνδέσεις μεταξύ των επεξεργαστών σε ένα δίκτυο με τον ίδιο τρόπο που ο ανθρώπινος εγκέφαλος αποθηκεύει και επεξεργάζεται τις πληροφορίες. Τα νευρωνικά δίκτυα προκαλούν μεγάλο ενδιαφέρον, έτσι ώστε πολλές επιχειρήσεις επενδύουν μεγάλα χρηματικά ποσά στην έρευνα και ανάπτυξή τους. Δυστυχώς, ενώ τα νευρωνικά δίκτυα επιτρέπουν την εκπαίδευση/ εμπλουτισμό της ΤΝ, είναι μια αργή διαδικασία.

5.4 Το πρόγραμμα WEKA

Το πλαίσιο εργασίας του Weka [9] περιέχει μια συλλογή των εργαλείων και των αλγορίθμων απεικόνισης για την ανάλυση στοιχείων και την πρόβλεψη μοντέλων, μαζί με τις γραφικές εφαρμογές για εύκολη πρόσβαση του χρήστη σε αυτή τη λειτουργία. Η αρχική έκδοση του Weka (προ Java) ήταν αλγόριθμοι μοντελοποίησης μεθόδου TCL/ TK front-end που εφαρμόστηκε σε άλλες γλώσσες προγραμματισμού, συν τις χρησιμότητες προ-επεξεργασίας στοιχείων στη C, και ένα σύστημα βασισμένο στην κατασκευή αρχείων για την επεξεργασία πειραμάτων εκμάθησης μηχανών. Αυτή η αρχική έκδοση, σχεδιάστηκε πρώτιστα ως εργαλείο για την ανάλυση στοιχείων από γεωργικές περιοχές [10] [11], αλλά η πιο πρόσφατη πλήρως βασισμένη στη Java έκδοση (Weka 3), για τις οποίες η ανάπτυξη άρχισε το 1997, χρησιμοποιείται τώρα σε πολλούς διαφορετικούς τομείς εφαρμογής, και ιδιαίτερα για εκπαιδευτικούς λόγους και την έρευνα. Τα κύρια χαρακτηριστικά του Weka είναι:

- ελεύθερα διαθέσιμο μέσω της GNU General Public License,
- πολύ εύκολο στη μεταφορά, επειδή εφαρμόζεται πλήρως στη γλώσσα προγραμματισμού της Java, και έτσι, τρέχει σχεδόν σε οποιαδήποτε σύγχρονη πλατφόρμα υπολογισμού,
- περιέχει μια περιεκτική συλλογή δεδομένων προ-επεξεργασίας και τεχνικών μοντελοποίησης, και
- είναι εύκολο να χρησιμοποιηθεί από έναν αρχάριο, λόγω των γραφικών εφαρμογών που διαθέτει.

Το Weka υποστηρίζει διάφορες τυποποιημένες εργασίες data mining, και πιο συγκεκριμένα, την προ-επεξεργασία δεδομένων, την κλασματοποίηση (clustering), την ταξινόμηση, την οπισθοδρόμηση, την απεικόνιση, και την επιλογή χαρακτηριστικών γνωρισμάτων. Όλες οι τεχνικές του Weka βασίζονται στην υπόθεση ότι τα δεδομένα είναι διαθέσιμα με τη μορφή απλού αρχείου ή σχέσεων, όπου κάθε σημείο δεδομένων περιγράφεται από έναν σταθερό αριθμό ιδιοτήτων (συνήθως, αριθμητικών ή ονομαστικών ιδιοτήτων, αλλά υποστηρίζονται επίσης και μερικοί άλλοι τύποι ιδιοτήτων). Το Weka παρέχει την πρόσβαση στις βάσεις

δεδομένων SQL χρησιμοποιώντας τη Java Database Connectivity και μπορεί να επεξεργαστεί το αποτέλεσμα που επιστρέφεται από μια αναζήτηση βάσεων δεδομένων. Δεν είναι ικανό να προβεί σε πολύ-συγγενικό data mining, αλλά υπάρχει χωριστό λογισμικό για τη μετατροπή μιας συλλογής συνδεμένων πινάκων βάσεων δεδομένων σε έναν ενιαίο πίνακα που είναι κατάλληλος για την επεξεργασία που χρησιμοποιεί το Weka [12]. Μια άλλη σημαντική περιοχή που δεν καλύπτεται από τους αλγορίθμους που περιλαμβάνονται στη διανομή του Weka, είναι η μοντελοποίηση ακολουθίας.

Η κύρια εφαρμογή διεπαφής με το χρήστη του Weka είναι ο Explorer, αλλά ουσιαστικά η ίδια λειτουργία μπορεί να προσεγγιστεί μέσω της εφαρμογής Knowledge Flow, και από τις γραμμές εντολών. Υπάρχει επίσης ο Experimenter, ο οποίος επιτρέπει τη συστηματική σύγκριση της προφητικής απόδοσης των μηχανών εκμάθησης αλγόριθμων του Weka με μια συλλογή των συνόλων δεδομένων.

Η εφαρμογή του Explorer έχει διάφορα επίπεδα πρόσβασης δεδομένων που δίνουν πρόσβαση στα κύρια συστατικά του πλαισίου εργασίας.

Η καρτέλα Preprocess παρέχει τις εγκαταστάσεις για την εισαγωγή των στοιχείων από μια βάση δεδομένων, ένα CSV αρχείο, κ.λπ., και για την προ-επεξεργασία αυτών των στοιχείων χρησιμοποιεί έναν αλγόριθμο φιλτραρίσματος. Αυτά τα φίλτρα μπορούν να χρησιμοποιηθούν για να μετασχηματίσουν τα στοιχεία (π.χ., να μετατρέψει αριθμητικές τιμές σε διακριτές) και έτσι δίνει τη δυνατότητα να διαγράφουν περιπτώσεις και ιδιότητες βάσει συγκεκριμένων κριτήριων.

Η καρτέλα Classify επιτρέπει στο χρήστη για να εφαρμόσει τους αλγορίθμους ταξινόμησης και οπισθοδρόμησης (οι οποίοι αποκαλούνται ταξινομητές- classifiers- στο Weka) στο προκύπτον σύνολο δεδομένων, για να υπολογίσει την ακρίβεια του πιθανού μοντέλου που προκύπτει, και να απεικονίσει τις λανθασμένες προβλέψεις, τις καμπύλες ROC, κ.λπ., ή το ίδιο το μοντέλο (εάν το μοντέλο είναι δυνατόν να απεικονιστεί όπως, π.χ., σε ένα δέντρο απόφασης).

Η καρτέλα Associate παρέχει την πρόσβαση στην εκμάθηση κανόνων που προσπαθούν να προσδιορίσουν τις πολύ σημαντικές αλληλεξαρτήσεις μεταξύ των ιδιοτήτων στα στοιχεία.

Η καρτέλα Cluster δίνει πρόσβαση στις τεχνικές clustering στο Weka, όπως, για παράδειγμα, ο απλός αλγόριθμος k-means. Επίσης, παρέχει μια εφαρμογή του αλγορίθμου μεγιστοποίησης της προσδοκίας.

Η επόμενη καρτέλα, Select attributes, παρέχει τους αλγορίθμους για την ταυτοποίηση των ιδιοτήτων που έχουν το μεγαλύτερο βαθμό πρόβλεψης σε ένα σύνολο δεδομένων.

Η τελευταία καρτέλα, η Visualize, παρουσιάζει έναν πίνακα πλοκών διασποράς, όπου οι μεμονωμένες πλοκές διασποράς μπορούν να επιλεχτούν και να διευρυνθούν, και να αναλυθούν περαιτέρω χρησιμοποιώντας διάφορες εφαρμογές χειρισμού.

5.5 Ιστορία του WEKA

- 1993 Το πανεπιστήμιο Waikato στη Νέα Ζηλανδία άρχισε την ανάπτυξη της αρχικής έκδοσης του Weka (που έγινε ένα μίγμα του TCL/ TK, της C, και της Makefiles).
- 1997 Λήφθηκε η απόφαση να ανακατασκευαστεί το Weka από την αρχή σε Java, συμπεριλαμβανομένων των εφαρμογών του διαμόρφωσης αλγορίθμων [13]
- 2005 Το Weka λαμβάνει το βραβείο SIGKDD για το Data Mining και το βραβείο Knowledge Discovery Service [14] [15]
- 2006 Η εταιρία Pentaho απόκτησε την αποκλειστική άδεια για να χρησιμοποιήσει το Weka για επιχειρηματική κατασκοπεία. Διαμορφώνεται η ανάσυρση δεδομένων (data mining) και το προφητικό σύστημα αναλύσεων της ακολουθίας επιχειρηματικής κατασκοπείας της Pentaho.

6. Ασφαλισιμότητα

Το πρόβλημα που αντιμετωπίζουμε σ' αυτή την εργασία αναφέρεται στην 'Ασφαλισιμότητα', δηλαδή στη λήψη της απόφασης κατά πόσο ένας υποψήφιος προς ασφάλιση, είναι αποδεκτού κινδύνου για να συναφθεί ασφαλιστήριο συμβόλαιο.

Ας αναλύσουμε όμως το πρόβλημα περαιτέρω.

Το πρόβλημά μας εστιάζεται στον έλεγχο των δεδομένων, των στοιχείων, ενός ατόμου που επιθυμεί να ξεκινήσει μία ασφάλεια ζωής, και πόσο ζημιογόνος θα μπορούσε ίσως να αποβεί σε μια ασφαλιστική εταιρία η ανάληψη του κινδύνου ασφάλισής του.

Μια απλή ασφάλιση ζωής γίνεται επί κάποιου ποσού που επιλέγει ο κάθε υποψήφιος (δηλαδή, μπορεί να κάνει κάποιος ασφάλεια ζωής ποσού 100.000,00 Ευρώ) και αυτό σημαίνει ότι σε περίπτωση θανάτου του (από ατύχημα, ασθένεια, ή ακόμα και αυτοκτονία) οι κληρονόμοι του θα πάρουν ως αποζημίωση το ποσό για το οποίο έχει γίνει η ασφάλιση.

Υπάρχουν όμως, πέραν της απλής ασφάλισης ζωής, και πρόσθετες ασφαλιστικές καλύψεις που θα μπορούσε να επιλέξει ο υποψήφιος, όπως Ιατροφαρμακευτική Περίθαλψη, ασφάλιση για Μόνιμη Ολική Ανικανότητα (αποζημίωση σε περίπτωση ατυχήματος που προκάλεσε μόνιμη ολική ανικανότητα), Πρόσκαιρη Ολική Ανικανότητα (αποζημίωση σε περίπτωση ατυχήματος που προκάλεσε πρόσκαιρη ολική ανικανότητα), Ασφάλιση Προσωπικού Ατυχήματος (αποζημίωση σε περίπτωση ατυχήματος που προκάλεσε απώλεια εισοδήματος λόγω ανικανότητας για εργασία), κτλ. Από τα παραπάνω, μεγαλύτερο ενδιαφέρον παρουσιάζει η Ασφάλιση Προσωπικού Ατυχήματος (ΠΑ), και γι' αυτόν τον λόγο το πρόγραμμά μας εστιάζει σε ασφάλειες ζωής με ΠΑ.

Για να μπορέσουμε όμως να εξετάσουμε την ασφαλιστική ικανότητα του υποψηφίου, θα πρέπει να έχουμε μερικές πληροφορίες για το προς ασφάλιση άτομο. Οι πληροφορίες αυτές θα πρέπει να μας βοηθήσουν να αποκτήσουμε μια εικόνα για το πόσο επιρρεπές σε ατυχήματα είναι το άτομο αυτό, πόσο συχνά θα χρειαστεί ιατρική βοήθεια κτλ. Με άλλα λόγια, θα πρέπει να έχουμε στα χέρια μας πληροφορίες σχετικά με την κατάσταση

της υγείας του (χρόνια νοσήματα), την ηλικία του (γηραιότερα ή πολύ νέα άτομα είναι πιο επιρρεπή σε ατυχήματα), το επάγγελμά του (κάποια επαγγέλματα είναι πιο επικίνδυνα από αλλά- παραδείγματος χάριν, πυροσβέστης), κτλ. Επίσης, επειδή γίνεται επιλογή πόσου ασφάλισης, για μεγάλα ποσά ασφάλισης απαιτούνται περισσότερες πληροφορίες απ' ότι για μικρότερα ποσά.

Συγκεκριμένα, για να μπορέσουμε να εκτιμήσουμε τον κίνδυνο ασφάλισης του εξεταζόμενου ατόμου, ζητάμε τις παρακάτω πληροφορίες:

1. Είδος ασφάλισης. Πραγματευόμαστε τρία είδη βασικών ασφαλειών:

    a. Απλή ασφάλεια ζωής ορισμένης διάρκειας.

    Είναι η απλή ασφάλεια ζωής που καλύπτει αποζημίωση σε περίπτωση θανάτου για τη διάρκεια ασφάλισης που έχει επιλεχθεί.

    b. Μικτή ασφάλεια ζωής με συμμέτοχη στα κέρδη.

    Αποτελεί συνδυασμό ασφάλειας ζωής και επενδυτικού προγράμματος. Απαιτεί επιλογή διάρκειας ασφάλισης, και ενώ καλύπτει αποζημίωση σε περίπτωση θανάτου, στη λήξη της αποδίδει και τα κέρδη από το επενδυτικό μέρος του προγράμματος.

    c. Ισόβια ασφάλεια ζωής.

    Είναι η απλή ασφάλεια ζωής που καλύπτει αποζημίωση σε περίπτωση θανάτου. Η διαφορά της από την απλή ασφάλεια ζωής ορισμένης διάρκειας είναι ότι δεν υπάρχει διάρκεια ασφάλισης, αλλά η ισχύς της είναι αόριστη.

    Στη μελέτη του προβλήματός μας, όλες οι παραπάνω ασφαλίσεις συνοδεύονται πάντα από την πρόσθετη κάλυψη του ΠΑ.

2. Γένος. Υπάρχει διαφοροποίηση για την ασφάλιση γυναικών και αντρών, ιδιαίτερα στον υπολογισμό του ετήσιου ασφαλίστρου

(είναι άλλωστε στατιστικά αποδεδειγμένο ότι οι γυναίκες ζουν περισσότερα χρόνια από τους άντρες).

3. Ηλικία. Όσο πιο νέος είναι ο υποψήφιος, τόσο πιο υγιής, άρα αποδεικνύεται και λιγότερο ζημιογόνος για την ασφαλιστική εταιρία. Επίσης, άτομα άνω των 65 δεν ασφαλίζονται, και οι ανήλικοι αντιμετωπίζονται ως άτομα 16 ή 20 ετών, ανάλογα με το είδος της βασικής ασφάλειας που επιλέγεται.

4. Ποσό ασφάλισης. Ανάλογα με το ύψος του ποσού ασφάλισης, είναι απαραίτητες ή όχι ορισμένες ιατρικές εξετάσεις για την εξακρίβωση της υγείας του υποψηφίου.

5. Διάρκεια ασφάλισης (όταν ζητείται ασφάλιση για συγκεκριμένο χρονικό διάστημα- περιπτώσεις Απλής ασφάλειας ζωής ορισμένης διάρκειας και Μικτής ασφάλειας ζωής με συμμέτοχη στα κέρδη).

6. Επάγγελμα. Μέσω του επαγγέλματος ελέγχεται η κλίμακα του κινδύνου στον οποίο εκτίθεται ο υποψήφιος στην καθημερινή του ζωή. Υπάρχουν λοιπόν τρείς κατηγορίες επαγγελμάτων, τα ακίνδυνα (πχ, τηλεφωνητές, ανθοπώλες, ηχολήπτες), τα μετρίου κινδύνου (πχ, αρτοποιοί, υδραυλικοί) και τα επικίνδυνα (πχ, δύτες, ναυτικοί). Η διαφορά που λαμβάνουμε υπόψη στο πρόβλημά μας, είναι ότι οι υποψήφιοι της πρώτης κατηγορίας ασφαλίζονται κανονικά, της δεύτερης ασφαλίζονται με κάποιο επασφάλιστρο (δηλαδή με κάποια προσαύξηση στα ασφάλιστρα σε σχέση με την πρώτη κατηγορία), ενώ όσοι υποψήφιοι ασκούν επαγγέλματα υψηλής επικινδυνότητας, δεν γίνονται αποδεκτοί για ασφάλιση από την ασφαλιστική εταιρία.

7. Ετήσιο εισόδημα.

8. Ποσό ασφάλισης ΠΑ. Ζητείται να επιλεγεί από τον υποψήφιο το ύψος του ποσού ασφάλισης ΠΑ, καθώς υπάρχουν ορισμένοι περιορισμοί για το ποσό, οι οποίοι όταν υπερβαίνονται, ο υποψήφιος κρίνεται μη ασφαλίσιμος.

9. Κατάσταση υγείας. Για περιπτώσεις όπως επιλογή πολύ μεγάλων ποσών ασφάλισης ζητείται από την ασφαλιστική

εταιρία η εκτίμηση της υγείας του υποψηφίου, και ανάλογα με τα αποτελέσματα των ιατρικών ελέγχων αποφασίζεται η ασφάλιση του ή όχι. Υπάρχουν τέσσερα διαφορετικά σύνολα εξετάσεων, από σύνολα με πιο απλές ιατρικές εξετάσεις, μέχρι και πιο σύνθετες. Σε ποιο σετ εξετάσεων θα υποβληθεί ο κάθε υποψήφιος, εξαρτάται από την ηλικία του και το επιλεγμένο ποσό βασικής ασφάλισης (όσο γηραιότερος και όσο μεγαλύτερο το ποσό, τόσο και πιο αναλυτικός ο ιατρικός έλεγχος που απαιτείται).

Όταν συλλεχθούν όλες οι απαραίτητες πληροφορίες για τον υποψήφιο προς ασφάλιση, γίνεται τελικά ο απολογισμός του κινδύνου ασφάλισής του. Ενώ αυτός ο έλεγχος συνήθως γίνεται από ένα έμπειρο στις ασφάλειες και στους κινδύνους που εμπλέκονται άτομο, εμείς θα προσπαθήσουμε να χρησιμοποιήσουμε διαφορετικά συστήματα για τη λήψη της απόφασης ασφάλισης.

Τα συστήματα που θα χρησιμοποιήσουμε βασίζονται σε ένα dataset 393 πραγματικών περιπτώσεων. Το σύνολο των περιπτώσεων δίδεται στο Παράρτημα Ι.

Η Ασφαλισιμότητα που αναφέρεται στον πίνακα του dataset αφορά το κατά πόσο σε κάθε περίπτωση έγινε Ασφάλιση (Α) ή όχι (Μ).

Για τη μελέτη του προβλήματος της Ασφαλισιμότητας, θα χρησιμοποιήσουμε τρεις διαφορετικές μεθόδους:

1. Ασαφή λογική με το εργαλείο Fuzzy CLIPS,
2. Κανόνες με συντελεστές βεβαιότητας (τύπου Mycin και τύπου Weighted) με το εργαλείο που υλοποιήθηκε στο [6],,
3. Νευρωνικά Δίκτυα με το εργαλείο WEKA.

Στην πρώτη περίπτωση (ασαφείς κανόνες), η σχεδίαση στηρίζεται στην χρήση εμπειρογνώμονα, ενώ στις δύο άλλες περιπτώσεις στη χρήση συνόλου εμπειρικών δεδομένων, από τα αρχεία ασφαλιστικής εταιρείας.

7. Υλοποίηση σε Fuzzy CLIPS

7.1 Γενικά

Για την υλοποίηση στο Fuzzy CLIPS, απαιτήθηκαν

- κανόνες ελέγχου των δεδομένων (για τον διαχωρισμό των αποδεκτών και μη αποδεκτών τιμών, τη λήψη αποφάσεων πυροδότησης των συναρτήσεων, την παραγωγή των αρχείων αποτελεσμάτων, κτλ), με σύνταξη:

  (defrule <name> [(salience k)] <condition>* => action*),

- συναρτήσεις (για την επεξεργασία των τιμών που προκύπτουν), με σύνταξη:

  (deffunction <name> (<variable name>*) (condition>*),

- κανόνες για τον καθορισμό της μορφής των γεγονότων , με σύνταξη:

  (deftemplate <name> (slot <slot name>)*).

Το πρόγραμμα δέχεται κατάλληλα διαμορφωμένο αρχείο δεδομένων, το αρχείο *arxikopoihsh_gegonotwn_A*, το οποίο δίδει για κάθε περίπτωση όλες τις απαραίτητες πληροφορίες για τη λήψη θετικής ή αρνητικής απόφασης ασφάλισης: το είδος ασφάλισης, το γένος του υποψηφίου, την ηλικία του, το επιλεγμένο ποσό βασικής ασφάλισης, το επάγγελμά του, το ετήσιο εισόδημά του, το ποσό ασφάλισης ΠΑ και τα αποτελέσματα των ιατρικών ελέγχων που απαιτούνται για την εξακρίβωση της υγείας του υποψηφίου.

Αφού οριστούν οι μεταβλητές, το πρόγραμμα ανοίγει το αρχείο των δεδομένων και δημιουργεί τα αρχεία αποτελεσμάτων *results_part1.txt* και *results_part2.txt*. Στη συνέχεια αρχίζει να πραγματοποιεί ελέγχους στα δεδομένα που έχουν εισαχθεί.

Η δομή του προγράμματος έχει ως εξής:

1. Ορισμός μεταβλητών

2. Σύνταξη δομημένων γεγονότων
3. Άνοιγμα αρχείων δεδομένων και αποτελεσμάτων
4. Αρχικοποίηση των μεταβλητών- έλεγχος αποδεκτών και μη τιμών
5. Καθορισμός των συνόλων δεδομένων με τις αποδεκτές τιμές
6. Υπολογισμός τιμολογήσεων βασικής ασφάλισης
7. Υπολογισμός βασικού ετήσιου ασφαλίστρου
8. Ορισμός ασαφών μεταβλητών
9. Ορισμός διαστημάτων ασαφών μεταβλητών
10. Τρέξιμο κανόνων αποφάσεων α΄ φάσης
11. Τρέξιμο κανόνων αποφάσεων β΄ φάσης
12. Μετατροπή fuzzy μεταβλητών σε crisp μεταβλητές
13. Υπολογισμός ασφαλίστρου ΠΑ
14. Αποτύπωση συμπερασμάτων στα αρχεία αποτελεσμάτων
15. Κλείσιμο προγράμματος.

Έτσι, πιο αναλυτικά, αρχικά το πρόγραμμα ελέγχει αν είναι αποδεκτή η ηλικία του υποψηφίου, και θέτει ως ελάχιστο όριο για απλή και την ισόβια βασική ασφάλιση την ηλικία των 16 ετών, ενώ για τη μικτή βασική ασφάλιση την ηλικία των 20 ετών. Στην περίπτωση λοιπόν που ζητάμε να ασφαλίσουμε ένα άτομο κάτω των ορίων αυτών, το πρόγραμμα θέτει ως ηλικία του υποψηφίου το κατώτερο σε κάθε περίπτωση όριο (αντίστοιχα 16 και 20 ετών) για να πραγματοποιηθούν όλοι οι έλεγχοι και οι υπολογισμοί. Επίσης, γενικά δεν ασφαλίζονται άτομα που έχουν ξεπεράσει το 65 έτος της ηλικίας τους.

Παράλληλα, το άθροισμα της ηλικίας κατά την έναρξη της ασφάλισης με τη επιλεγμένη διάρκεια ασφάλισης δεν πρέπει να υπερβαίνει το 65,

έλεγχος που πραγματοποιείται στη συνέχεια. Αν υπερβαίνεται το 65, τότε τίθεται ως διάρκεια ασφάλισης τα (65- ηλικία υποψηφίου) έτη.

Ακόμα, το πρόγραμμα ελέγχει την επικινδυνότητα του επαγγέλματος του υποψηφίου, και τον κατατάσσει σε μια από τις τρεις κλάσεις επικινδυνότητας: τα ακίνδυνα, τα ελαφρώς επικίνδυνα και τα επικίνδυνα επαγγέλματα. Από αυτά, τα επαγγέλματα που κατηγοριοποιούνται στις δυο πρώτες κλάσεις είναι αποδεκτά προς ασφάλιση, ενώ όσα κατηγοριοποιούνται στην τρίτη, απορρίπτονται αυτομάτως, και η ασφαλιστική εταιρία δεν αναλαμβάνει την ανάληψη κινδύνου ασφάλισης. Τα επαγγέλματα της πρώτης κλάσης είναι τα εξής:

- Αισθητικοί, Ανθοπώλες, Αποθηκάριοι, Αρχαιολόγοι, Αρχιτέκτονες, Ασφαλιστές, Βιβλιοθηκάριοι, Βιβλιοπώλες, Βιομήχανοι, Βιοτέχνες, Βιοχημικοί, Γεωλόγοι, Γεωπόνοι, Γλύπτες, Γουνοποιοί, Γραφίστες, Γυμναστές, Διαιτητές, Διακοσμητές, Διαφημιστές, Δικαστές- Δικηγόροι, Εισπράκτορες, Εκδότες, Εκπαιδευτικοί, Έμποροι ρουχισμού, Εστιάτορές, Εφοπλιστές, Ζωγράφοι, Ηθοποιοί, Ηλεκτρολόγοι- Μηχανολόγοι, Ηχολήπτες, Θυρωροί, Ιατροί, Ιερείς, Ιχθυοπώλες, Καθαριστήρια- Βαφεία, Καθαρίστριες, Καθηγητές, Καφεπώλες, Γραφεία Κηδειών, Κηπουροί, Κομμωτές- Κουρείς, Κοσμηματοπώλες, Λαχειοπώλες, Λογιστές, Μαθητές, Μετερεολόγοι, Μοδίστρες, Μοντέλα, Μουσικοί, Μπάρμαν, Νηπιαγωγοί, Νοσοκόμοι, Οδοκαθαριστές, Οδοντοτεχνίτες, Οικοκυρές, Πρατηριούχοι Τσιγάρων, Πολιτικοί Μηχανικοί, Προπονητές, Πωλητές, Ράπτες, Σερβιτόροι, Σκηνοθέτες, Συμβολαιογράφοι, Συντηρητές Έργων Τέχνης, Σχεδιαστές, Ταπετσιέρηδες, Τηλεφωνητές, Τοπογράφοι, Τραγουδιστές, Τυπογράφοι, Υπάλληλοι Γραφείου, Φαρμακοποιοί, Φοιτητές, Φυσιοθεραπευτές, Φωτογράφοι, Χημικοί, Χρηματιστές, Ωρολογοποιοί,

τα οποία ασφαλίζονται κανονικά.

Τα επαγγέλματα της δεύτερης κλάσης είναι τα εξής:

- Αγρότες, Αρτοποιοί, Γαλακτοκόμοι, Δασοφύλακες, Δημοσιογράφοι, Ελαιοχρωματιστές, Έμποροι κατασκευαστικών υλικών, Επιπλοποιοί, Ζαχαροπλάστες, Ηλεκτρολόγοι- Ηλεκτρονικοί, Κτηνοτρόφοι- Πτηνοτρόφοι, Κτίστες, Λατόμοι, Λιμενοφύλακες, Μάγειρες, Μαργαρώδες- Πλακάδες, Μελισσοκόμοι, Μεταλλουργοί,

Μεταφορείς, Μηχανικοί Αυτοκινήτων, Ναυαγοσώστες, Ντετέκτιβ, Νυχτοφύλακες, Επαγγελματίες οδηγοί, Οικοδόμοι, Οπωροπώλες, Πρατηριούχοι υγρών καυσίμων, Πυροσβέστες, Συντηρητές καυστήρων, Τορναδόροι, Τυροκόμοι, Υαλουργοί, Υδραυλικοί, Υποδηματοποιοί, Υφαντουργοί, Φανοποιοί, Φύλακες, Χειριστές μηχανημάτων, Χορευτές, Ψητοπώλες, Ψυκτικοί,

τα οποία ασφαλίζονται, αλλά με κάποια επιβάρυνση στα ασφάλιστρα.

Τα επαγγέλματα της τρίτης κλάσης είναι τα εξής:

- Αθλητές, Ανθρακωρύχοι, Δύτες, Οδηγοί αγώνων, Ναυτικοί , Πιλότοι, Στρατιωτικοί,

τα οποία και δεν ασφαλίζονται λόγω της μεγάλης επικινδυνότητας του επαγγέλματος του υποψηφίου.

Στη συνέχεια, ελέγχεται αν απαιτούνται ιατρικές ανάλογα με το επιλεγμένο ποσό βασικής ασφάλισης και την ηλικία του υποψηφίου, και τι εξετάσεις απαιτούνται. Συγκεκριμένα,

- εάν ισχύει
    - ποσό ασφάλισης >= 22.500 Ευρώ ή
    - 16 <=ηλικία <= 55 και 22.500<= ποσό ασφάλισης <=30.000 ή
    - 16 <=ηλικία <= 45 και 30.000<= ποσό ασφάλισης <=45.000

    Δεν απαιτούνται ιατρικές εξετάσεις, και αρκεί υπεύθυνη δήλωση του υποψηφίου ή του κηδεμόνα του εάν πρόκειται για ανήλικο ασφαλιζόμενο.

- Για
    - 16 <=ηλικία <= 45 και 45.000<= ποσό ασφάλισης <=60.000 ή
    - 46 <=ηλικία <= 55 και 30.000<= ποσό ασφάλισης <=45.000 ή
    - 56 <=ηλικία <= 65 και 22.500<= ποσό ασφάλισης <=30.000

    Απαιτούνται

1. Παθολογική εξέταση,
2. Ηλεκτροκαρδιογράφημα, και
3. Ακτινογραφία θώρακος.

- Για
  - 56 <=ηλικία <= 65 και 30.000<= ποσό ασφάλισης <=45.000 ή
  - 16 <=ηλικία <= 45 και 60.000<= ποσό ασφάλισης <=190.000

  Απαιτούνται
  1. Παθολογική εξέταση,
  2. Ηλεκτροκαρδιογράφημα,
  3. Ακτινογραφία θώρακος,
  4. Ανάλυση ούρων, και
  5. Γενική αίματος.
- Για
  - 46 <=ηλικία <= 65 και 45.000<= ποσό ασφάλισης <= 190.000

  Απαιτούνται
  1. Παθολογική εξέταση,
  2. Ηλεκτροκαρδιογράφημα,
  3. Ακτινογραφία θώρακος,
  4. Ανάλυση ούρων,
  5. Γενική αίματος,
  6. Σάκχαρο αίματος,
  7. Χοληστερίνη- HDL, και

8. Τριγλυκερίδια.

- Για
  - ποσό ασφάλισης >= 190.000

  Απαιτούνται

  1. Παθολογική εξέταση,
  2. Ηλεκτροκαρδιογράφημα,
  3. Ακτινογραφία θώρακος,
  4. Ανάλυση ούρων,
  5. Γενική αίματος,
  6. Σάκχαρο αίματος,
  7. Χοληστερίνη- HDL,
  8. Τριγλυκερίδια,
  9. Τεστ κοπώσεως,
  10. Εξέταση για AIDS, και
  11. Ταχύτητα καθίζησης.

Αφού γίνουν οι απαιτούμενες εξετάσεις, τα αποτελέσματα των οποίων πρέπει να δοθούν σε μια κλίμακα από το 0 ως το 5 (5- Άριστες, 4- Πολύ καλές, 3- Καλές, 2- Μέτριες, 1-Κακές, και 0-Πολύ κακές), υπολογίζεται η μεταβλητή της υγείας, η οποία μας καθορίζει κατά πόσο, από άποψη υγείας, ο υποψήφιος είναι κατάλληλος προς ασφάλιση. Η συνάρτηση που χρησιμοποιούμε για τον υπολογισμό την μεταβλητής της υγείας εξαρτάται από ποιο σετ εξετάσεων υποχρεούται ο υποψήφιος να καλύψει. Όταν λοιπόν του ζητείται να υποβληθεί στο πρώτο σετ εξετάσεων, η μεταβλητή της υγείας υπολογίζεται από τη σχέση

Υγεία = 0.2 * (0.35 * t1 + 0.35 * t2 + 0.3 * t3),

όταν ζητείται να υποβληθεί στο δεύτερο σετ εξετάσεων

Υγεία = 0.2 * (0.2 * t1 + 0.25 * t2 + 0.19 * t3 + 0.18 * t4 + 0.18 * t5),

όταν ζητείται να υποβληθεί στο τρίτο σετ εξετάσεων

Υγεία = 0.2 * (0.13 * t1 + 0.13 * t2 + 0.1 * t3 + 0.12 * t4 + 0.13 * t5 + 0.13 * t6 + 0.13 * t7 + 0.13 * t8),

όταν ζητείται να υποβληθεί στο τέταρτο σετ εξετάσεων

Υγεία = 0.2 * (0.08 * t1 + 0.1 * t2 + 0.08 * t3 + 0.08 * t4 + 0.08 * t5 + 0.07 * t6 + 0.1 * t7 + 0.1 * t8 + 0.1 * t9 + 0.15 * t10 + 0.06 * t11),

ενώ όταν δεν απαιτούνται ιατρικές εξετάσεις, η μεταβλητή της υγείας λαμβάνει την τιμή μηδέν. Τα t1, t2, …, t11 που εμφανίζονται παραπάνω, αντιστοιχούν στις βαθμολογήσεις που έχουν θέσει οι αντίστοιχοι ιατροί στην $1^{η}$, $2^{η}$, …, $11^{η}$ εξέταση του συνόλου των εξετάσεων που απαιτούνται.

Να σημειώσουμε, ότι ενώ μιλάμε για FuzzyCLIPS οι μεταβλητές που δίνουμε ως τώρα είναι διακριτές. Αυτό οφείλεται στο ότι υπολογίζουμε αρχικά τις διακριτές τιμές των μεταβλητών , και στη συνέχεις τις μεταφράζουμε σε μεταβλητές με αβεβαιότητα και οι υπολογισμοί μας όλοι βασίζονται στις αβέβαιες μεταβλητές.

Ο επόμενος έλεγχος που πραγματοποιείται είναι ο έλεγχος του επιλεγμένου ποσού ασφάλισης ΠΑ. Έτσι, το ύψος του ποσού ασφάλισης ΠΑ δεν μπορεί να υπερβαίνει:

- τις 200.000,00 Ευρώ, ή
- το τετραπλάσιο του βασικού ασφαλίστρου ζωής, ή
- το ετήσιο εισόδημά του.

Σε αυτές τις τρείς περιπτώσεις, θεωρείται ότι υπάρχει δόλος και η αίτηση ασφάλισης για ΠΑ καθίσταται αυτομάτως μη αποδεκτή. Επίσης, ανάλογα με το ποσό ασφάλισης ΠΑ είναι και το ασφάλιστρο για κάλυψη ΠΑ. Αν μετά από αυτούς τους ελέγχους δεν υπάρχει υπέρβαση των όρων ασφάλισης, το πρόγραμμα προχωράει κανονικά στον υπολογισμό των ασφαλίστρων ΠΑ και την μελέτη της ασφαλισιμότητας.

Στην πορεία, το πρόγραμμα υπολογίζει τα ασφάλιστρα για τη βασική ασφάλιση ζωής, ανάλογα με το είδος βασικής ασφάλισης που έχει επιλεγεί,

και μετά εισάγεται το Fuzzy μέρος του προγράμματος, με τους κανόνες ελέγχου καταλληλότητας ασφάλισης.

Η απόφαση ασφάλισης πραγματοποιείται σε δύο φάσεις. Στην πρώτη λαμβάνονται υπόψη οι μεταβλητές ηλικία, ποσό ασφάλισης και εισόδημα για να γίνει ένας πρώτος υπολογισμός καταλληλότητας ασφάλισης. Στη δεύτερη χρησιμοποιείται το αποτέλεσμα της πρώτης φάσης μαζί με τις μεταβλητές ηλικία και υγεία για να παραχθεί το τελικό αποτέλεσμα ασφάλισης.

Σε αυτή τη φάση έχει ήδη υπολογιστεί η καταλληλότητα ασφάλισης, και το υπόλοιπο πρόγραμμα ασχολείται με τον υπολογισμό του ασφαλίστρου ΠΑ και να προβεί τελικά στον υπολογισμό των συνολικών ασφαλίστρων. Η τιμολόγηση του ΠΑ βασίζεται στο βαθμό επικινδυνότητας που προκύπτει από το επάγγελμα του υποψηφίου και την ηλικία του (όσο πιο νέος ο υποψήφιος τόσο πιο προσιτά τα ασφάλιστρα).

Όσο για το συνολικό ετήσιο ασφάλιστρο ζωής, αυτό προκύπτει συναρτήσει της μεταβλητής υγείας:

- Αν η τιμή της είναι υψηλή, δηλαδή αν υπερβαίνει το 80, τα συνολικά ασφάλιστρα είναι το άθροισμα του ασφάλιστρου βασικής ασφάλισης και του ασφάλιστρου ΠΑ.

- Αν η τιμή της μεταβλητής της υγείας κυμαίνεται μεταξύ 65 και 80, τα συνολικά ασφάλιστρα είναι το άθροισμα του ασφάλιστρου βασικής ασφάλισης και του ασφάλιστρου ΠΑ, προσαυξημένα κατά 2%.

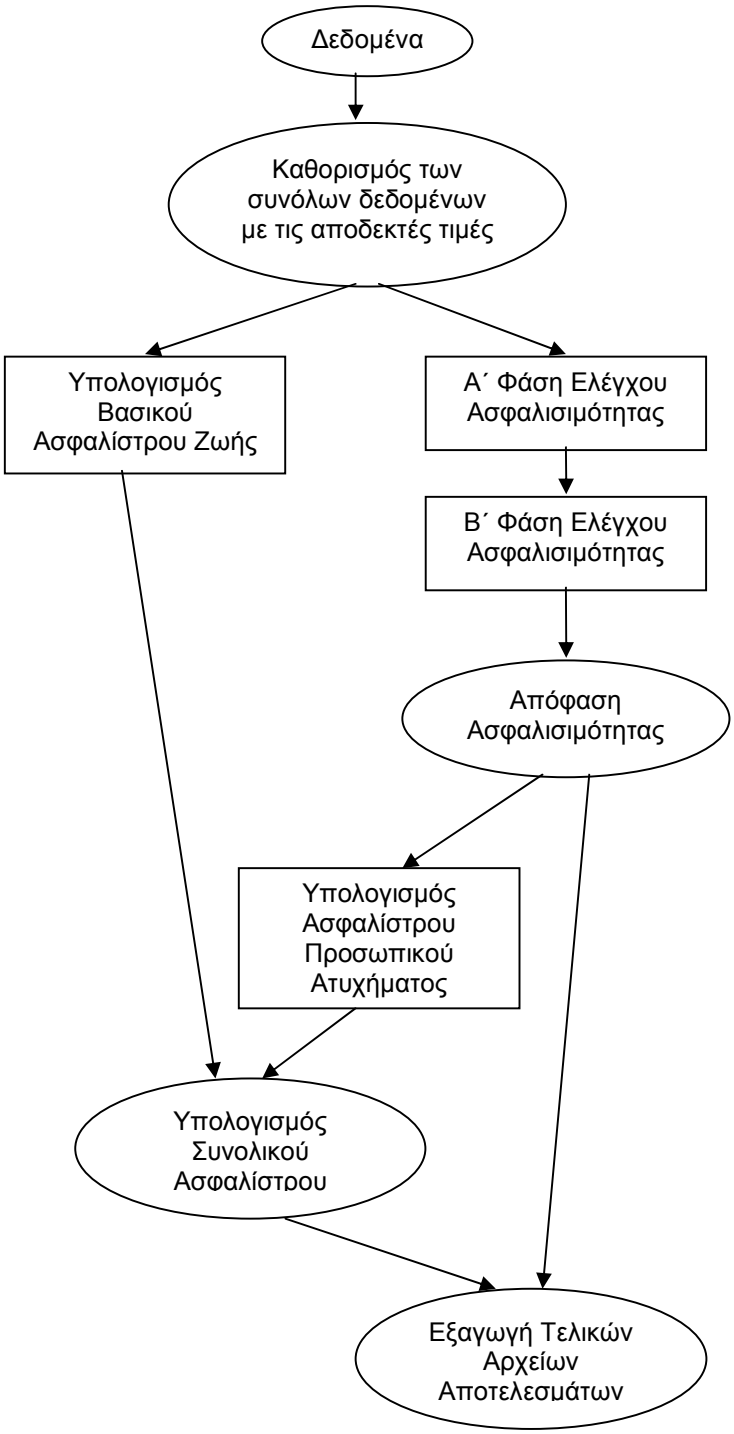
- Αν η τιμή της μεταβλητής της υγείας κυμαίνεται μεταξύ 50 και 65, τα συνολικά ασφάλιστρα είναι το άθροισμα του ασφάλιστρου βασικής ασφάλισης και του ασφάλιστρου ΠΑ, προσαυξημένα κατά 6%.

- Αν η τιμή της μεταβλητής της υγείας κυμαίνεται μεταξύ 40 και 50, τα συνολικά ασφάλιστρα είναι το άθροισμα του ασφάλιστρου βασικής ασφάλισης και του ασφάλιστρου ΠΑ, προσαυξημένα κατά 10%.

- Αν η τιμή της μεταβλητής της υγείας κυμαίνεται μεταξύ 0 και 40, η ασφαλιστική εταιρία δεν αναλαμβάνει την ασφάλιση λόγω υψηλού κινδύνου, και το πρόγραμμα θέτει ως τιμή συνολικών ασφάλιστρων μηδέν.

- Αν ο υποψήφιος ασκεί ένα από τα επαγγέλματα μεγάλης επικινδυνότητας, η τιμή της μεταβλητής υγείας έχει τεθεί ίση με μηδέν, και τα συνολικά ετήσια ασφάλιστρα μηδενίζονται αυτομάτως.

Μια σχηματική αναπαράσταση της λειτουργίας του προγράμματος φαίνεται στο Σχήμα 5.



Σχήμα 5. Σχηματική Αναπαράσταση Λειτουργίας προγράμματος σε Fuzzy CLIPS.

Όπως είδαμε λοιπόν, με το πρόγραμμα αυτό, δεν γίνεται απλά έλεγχος της Ασφαλισιμότητας του υποψηφίου, άλλα γίνεται παράλληλα και υπολογισμός των συνολικών ετήσιων ασφαλίστρων (βασικό ετήσιο ασφάλιστρο + ασφάλιστρο ΠΑ). Το βασικό όμως πρόβλημά μας, και είναι αυτό στο οποίο εστιάζουμε και στις επόμενες εφαρμογές, είναι ο καθορισμός της Ασφαλισιμότητας.

7.2 Καθορισμός training- test set

Για να μπορέσουμε να έχουμε μια εικόνα του κατά πόσο το σύστημά μας λειτουργεί ικανοποιητικά, χωρίζουμε το σύνολο των δεδομένων μας δε δυο set δεδομένων με αναλογία 2 προς 1, τα training και test set, αντίστοιχα. Στο πρόβλημα της Ασφαλισιμότητας, για να μπορέσουμε να έχουμε πιο αντικειμενικά αποτελέσματα προσπαθήσαμε να πάρουμε ίσους αριθμούς περιπτώσεων από κάθε είδος ασφάλισης για τη δημιουργία του test set (γενικά το δείγμα μας αποτελείται από περίπου ίσες περιπτώσεις από κάθε είδος ασφάλειας), ενώ οι περιπτώσεις που απομένουν αποτελούν το training set μας.

Έτσι το training set αποτελείται από 270 περιπτώσεις από όλα τα είδη ασφαλειών, ενώ το test set από 123 περιπτώσεις αντίστοιχα.

7.3 Μετρικές

Για να ελέγξουμε την καταλληλότητα του προγράμματος θα χρησιμοποιήσουμε τέσσερις μετρικές, τιμές των οποίων θα υπολογίσουμε τόσο από τα αποτελέσματα του training set όσο και από αυτά του test set. Αυτές είναι οι:

- Η Ακρίβεια (Accuracy): μας δηλώνει το ποσοστό των γεγονότων του συνόλου μας για τα οποία έγινε σωστή πρόβλεψη,
- Η Ευαισθησία (Sensitivity): μας δηλώνει το ποσοστό των θετικών γεγονότων για τα οποία έγινε σωστή πρόβλεψη,
- Η Ιδιαιτερότητα (Specificity): μας δηλώνει το ποσοστό των αρνητικών γεγονότων για τα οποία έγινε σωστή πρόβλεψη,
- Η Ακρίβεια (Precision): μας δηλώνει το ποσοστό των θετικών προβλέψεων που ήταν στην πραγματικότητα θετικά γεγονότα.

Οι τύποι τους είναι οι εξής:

$$Accuracy = (a+d)/(a+b+c+d),$$

$$Sensitivity = a/(a+b),$$

$$Specificity = d/(d+c),$$

$$Precision = a/(a+c).$$

Όπου:

- **a**: πλήθος περιπτώσεων που ανήκουν σε μια κλάση C και ταξινομήθηκαν σ' αυτή,
- **b**: πλήθος περιπτώσεων που ανήκουν σε μια κλάση C και δεν ταξινομήθηκαν σ' αυτή,
- **c**: πλήθος περιπτώσεων που δεν ανήκουν σε μια κλάση C και ταξινομήθηκαν σ' αυτή,
- **d**: πλήθος περιπτώσεων που δεν ανήκουν σε μια κλάση C και δεν ταξινομήθηκαν σ' αυτή.

Οι μετρικές αυτές δεν εντάσσονται στο αρχικό μας πρόγραμμα, αλλά είναι ένα ξεχωριστό πρόγραμμα σε CLIPS, το οποίο χρησιμοποιεί ως δεδομένα τα αρχεία αποτελεσμάτων του κυρίως προγράμματος, στο οποίο αναφέρονται η τιμή της εκτίμησης της Ασφαλισιμότητας και η τιμή της πραγματικής ασφάλισης ή μη του υποψηφίου, από τις οποίες υπολογίζει τις a, b, c και d.

Μετά από την εφαρμογή των αρχείων αποτελεσμάτων στο πρόγραμμα υπολογισμού των μετρικών, παίρνουμε τις παρακάτω τιμές:

Training Set		**Test Set**	
Accuracy	0.7426	Accuracy	0.7886
Sensitivity	0.9394	Sensitivity	0.9109
Specificity	0.2162	Specificity	0.1905
Precision	0.7623	Precision	0.844

Πίνακας 4. Αποτελέσματα μετρικών εφαρμογής FuzzyCLIPS.

Παρατηρούμε λοιπόν, ότι οι μετρικές Accuracy και Precision έχουν βελτιωθεί αισθητά (ιδιαίτερα η Precision) στο test set, ενώ οι Sensitivity και Specificity, έχουν μια μικρή πτώση. Αυτό σημαίνει, ότι το πρόγραμμα αναγνωρίζει και κατηγοριοποιεί σωστά τις προς ασφάλιση περιπτώσεις, ενώ έχει μια μικρή απόκλιση στις ορθές προβλέψεις ασφαλίσιμων ή μη περιπτώσεων. Αποδεικνύεται δε, ότι το πρόγραμμα έχει πολύ καλή απόδοση στην ορθή κατηγοριοποίηση ως ασφαλίσιμων περιπτώσεων (περίπου της τάξης του 91-94%), ενώ πολύ κακή απόδοση στην ορθή κατηγοριοποίηση ως μη ασφαλίσιμων περιπτώσεων (19-22%).

8. Υλοποίηση με το Εργαλείο Παραγωγής Έμπειρων Συστημάτων με Συντελεστές Βεβαιότητας

Εξετάζοντας πάντα το πρόβλημα της ασφαλισιμότητας, για να μπορέσουμε να εφαρμόσουμε τη Μέθοδο Συντελεστών βεβαιότητας με το προαναφερθέν εργαλείο, πρέπει να προσαρμόσουμε το σύνολο των δεδομένων μας, ώστε να γίνονται αποδεκτά από το εργαλείο.

Έτσι, κατασκευάζουμε δυο αρχεία εισόδου, το dataset.data και το settings.txt. Αυτά έχουν την δομή:

<αποτέλεσμα>,<τιμή-παραμέτρου-1>,<τιμή-παραμέτρου-2>, …, <τιμή-παραμέτρου-ν>

και,

<όνομα έμπειρου συστήματος>

<πλήθος παραμέτρων συστήματος>

<πλήθος παραμέτρων πρώτης φάσης>

<αριθμοί παραμέτρων πρώτης φάσης>

<πλήθος παραμέτρων δεύτερης φάσης>

<αριθμοί παραμέτρων πρώτης φάσης>

<όνομα παραμέτρου εξόδου>

<όνομα παραμέτρου εισόδου-1>

<όνομα παραμέτρου εισόδου-2>

…

<όνομα παραμέτρου εισόδου-ν>

αντίστοιχα, άρα στο πρόβλημά μας τα αρχεία θα είναι της μορφής: (dataset.data)

Old,Medium,MediumHigh,Old,None,yes
Young,VeryLow,LowMedium,Young,None,yes
Young,Low,LowMedium,Young,None,yes
Young,VeryLow,LowMedium,Young,None,yes
Young,Low,LowMedium,Young,None,yes
Young,VeryLow,LowMedium,Young,None,yes
Young,Medium,MediumHigh,Young,None,yes
Young,Low,LowMedium,Young,None,yes
Old,Low,LowMedium,Old,None,yes
Young,Low,LowMedium,Young,None,yes
Young,VeryLow,Low,Young,None,yes
Old,Low,LowMedium,Old,None,yes
VeryOld,Low,MediumHigh,VeryOld,None,no
Young,Low,MediumHigh,Young,None,yes
Young,Medium,MediumHigh,Young,None,yes
Old,Low,MediumHigh,Old,None,no
Old,High,MediumHigh,Old,None,yes
Young,Medium,MediumHigh,Young,None,no
…….

και (settings.txt)

AsfalishZwhs
6
3
1,2,3
2
4,5
hlikia1
poso_asfalishs
eisodhma
hlikia2
ygeia
Asfalisimothta

Το αρχείο settings.txt δηλαδή, στην ουσία καθορίζει τη μορφή που θα έχει το αρχείο dataset.data.

Και σε αυτήν την εφαρμογή, βλέπουμε ότι ο έλεγχος γίνεται σε δυο φάσεις, παρόμοιες με αυτές που ορίσαμε στο FuzzyCLIPS. Στην πρώτη φάση, γίνεται έλεγχος με βάση τις τιμές των μεταβλητών ηλικία, ποσό

ασφάλισης και το εισόδημα, ενώ στη δεύτερη, βάσει την ηλικία και την υγεία του υποψηφίου (Σημειώνεται ότι οι μεταβλητές hlikia1 και hlikia2 είναι στην ουσία η ίδια μεταβλητή της ηλικίας).

Ακόμα, οι τιμές που εμφανίζονται στο dataset.data, είναι οι τιμές των παραπάνω μεταβλητων, οι οποίες έχουν χωριστει στις κλάσεις:

1. hlikia1: VeryYoung, Young, Old, VeryOld.
2. poso_asfalishs: VeryLow, Low, Medium, High, VeryHigh.
3. eisodhma: Low, LowMedium, MediumHigh, High.
4. hlikia2: VeryYoung, Young, Old, VeryOld.
5. ygeia: Bad, Medium, Good, None.
6. Asfalisimothta: yes,no.

οι οποίες αντιστοιχούν σε τιμές όπως αναλύονται στους Πίνακες 5- 8.

ΗΛΙΚΙΑ			
VeryYoung	16	-	22
Young	23	-	40
Old	41	-	55
VeryOld	56	-	65

Πίνακας 5. Κλίμακες ηλικίας.

ΠΟΣΟ ΒΑΣΙΚΗΣ ΑΣΦΑΛΙΣΗΣ ΖΩΗΣ				
1	VeryLow	200	-	1700
2	Low	1701	-	6000
3	Medium	6001	-	20000
4	High	2001	-	45000
5	VeryHigh	45001	-	110000

Πίνακας 6. Κλίμακες ποσού βασικής ασφάλισης.

ΕΤΗΣΙΟ ΕΙΣΟΔΗΜΑ				
1	Low	2000	-	4500
2	LowMedium	4501	-	15000
3	MediumHigh	15001	-	45000
4	High	45001	-	110000

Πίνακας 7. Κλίμακες ύψους ετησίου εισοδήματος.

ΥΓΕΙΑ (Score)				
1	Bad	0.00	-	0.40
2	Medium	0.41	-	0.65
3	Good	0.66	-	1.00

Πίνακας 8. Κλίμακες αξιολόγησης υγείας.

Και πάλι χρησιμοποιούμε τα training και test set που χρησιμοποιήσαμε και στο FuzzyCLIPS, κατάλληλα διαμορφωμένα, για να μπορέσουμε στην πορεία να συγκρίνουμε τα αποτελέσματα που παράγονται.

Η ίδια η εφαρμογή λειτουργεί συνοπτικά ως εξής: το data set χωρίζεται σε training set και test set κατά πέντε διαφορετικούς τρόπους, διατηρώντας όμως τη σχέση 2/1, και για κάθε περίπτωση βρίσκονται οι κανόνες και οι συντελεστές βεβαιότητας της πρώτης (First Group) και δεύτερης φάσης (Second Group). Δημιουργείται το αντίστοιχο πρόγραμμα CLIPS, υπολογίζονται οι παράμετροι w1, w2 και w μέσω του Γενετικού Αλγορίθμου, και στη συνέχεια δοκιμάζονται τα δύο συστήματα (MYCIN, Weighted) στο test set που παρήχθη, και υπολογίζονται οι τιμές των μετρικών Accuracy, Sensitivity, Specificity και Precision).

Οι τιμές αυτές είναι οι ίδιες οι μετρικές που υπολογίσαμε και στο FuzzyCLIPS, και θα τις χρησιμοποιήσουμε για να συγκρίνουμε την αποτελεσματικότητα των δυο μεθόδων.

Αφού τρέξουμε το πρόγραμμα, οι τιμές των μετρικών που λαμβάνουμε είναι αυτές που φαίνονται στους Πίνακες 9 και 10 για τα συστήματα MYCIN και Weighted, αντίστοιχα.

MYCIN			
Training Set		**Test Set**	
Accuracy	0.256818	Accuracy	0.154386
Sensitivity	0.0	Sensitivity	0.121212
Specificity	0.982609	Specificity	0.93333
Precision	-1	Precision	-1

Πίνακας 9. Αποτελέσματα μετρικών Υλοποίηση της Μεθόδου Συντελεστών βεβαιότητας σε Έμπειρο Σύστημα-MYCIN.

Weighted			
Training Set		**Test Set**	
Accuracy	0.672727	Accuracy	0.804184
Sensitivity	0.744615	Sensitivity	0.87803
Specificity	0.469565	Specificity	0.4
Precision	0.801677	Precision	0.888885

Πίνακας 10. Αποτελέσματα μετρικών Υλοποίηση της Μεθόδου Συντελεστών βεβαιότητας σε Έμπειρο Σύστημα- Weighted.

Παρατηρούμε λοιπόν, ότι ενώ η μετρική Specificity παρουσιάζει πολύ καλές τιμές για τη μέθοδο με τη χρήση του MYCIN, οι υπόλοιπες μετρικές είναι ιδιαίτερα χαμηλές, ως και μη αποδεκτές, τόσο στο training set, όσο και στο test set.

Αντιθέτως, στο σύστημα Weighted η μετρική Specificity δεν παρουσιάζει ιδιαίτερα μεγάλες τιμές, οι τιμές όμως των υπολοίπων μετρικών είναι κατά πολύ καλύτερες από αυτές του συστήματος σε MYCIN, και ειδικά στο test set προσεγγίζουν το 89% (από 80-89%).

Σε γενικές γραμμές, οι τιμές των μετρικών του συστήματος Weighted είναι σαφώς καλύτερες στο σύνολό τους από αυτές του συστήματος σε MYCIN. Στη συνέχεια αυτές τις τιμές χρησιμοποιούμε για τη σύγκριση με τις άλλες μεθόδους.

9. Εφαρμογή σε WEKA

Όπως και στις προηγούμενες δυο εφαρμογές, έτσι κι εδώ, για να τρέξουμε το πρόγραμμα Weka θα πρέπει να δώσουμε τα δεδομένα μας στην κατάλληλη μορφή. Η δομή του αρχείου εισόδου θα πρέπει να είναι ως εξής:

@relation AsfaliesZwhs_Fash2

@attribute hlikia2 {VeryYoung, Young, Old, VeryOld}
@attribute ygeia {Bad, Medium, Good, None}
@attribute Asfalisimothta {yes, no}

@data
Old,None,yes
Young,None,yes
Young,None,yes
Young,None,yes
Young,None,yes
Young,None,yes
Young,None,yes
Young,None,yes
Old,None,yes
Young,None,yes
Young,None,yes
Old,None,yes
VeryOld,None,no
Young,None,yes
…..

όπου στην πρώτη γραμμή

@relation AsfaliesZwhs_Fash2

δηλώνουμε το όνομα του προβλήματός μας, στη συνέχεια, με τις εντολές

@attribute hlikia2 {VeryYoung, Young, Old, VeryOld}

δηλώνουμε τα ονόματα των χαρακτηριστικών και τον τύπο τους (στο πρόβλημά μας τα χαρακτηριστικά παίρνουν μια από τις τιμές που δηλώνονται), ενώ η τελευταία από αυτές τις εντολές αποτελεί τη δήλωση των κλάσεων προς αναγνώριση. Τέλος, ακολουθούν οι εντολές της μορφής

Old,None,yes
Young,None,yes
Young,None,yes…

οι οποίες δηλώνουν τις τιμές του κάθε χαρακτηριστικού για τα δεδομένα μας (κάθε γραμμή αντιπροσωπεύει και μια περίπτωση), και διαχωρίζεται από τα υπόλοιπα στοιχειά με τη δήλωση @data πριν τις τιμές.

Αφού κατασκευάζουμε το αρχείο εισόδου των δεδομένων μας, αρκεί να ρυθμίσουμε το πρόγραμμα Weka, επιλέγοντας τον αλγόριθμο μάθησης που επιθυμούμε και τις παραμέτρους για τις οποίες θα τρέξουμε την εφαρμογή. Στο συγκεκριμένο πρόβλημα, χρησιμοποιήσαμε ρυθμό μάθησης ίσο με 0.3, σταθερά ορμής ίση με 0.2 και 500 κύκλους εκπαίδευσης.

Ακόμα, στο πρόβλημα της ασφαλισιμότητας χρησιμοποιήσαμε τρία διαφορετικά αρχεία δεδομένων των 393 περιπτώσεων, εξετάζοντας σε κάθε αρχείο και διαφορετικά χαρακτηριστικά. Τα τρία αυτά αρχεία είναι:

- *AsfaliesZwhs_Fash1.arff*, όπου τα χαρακτηριστικά που χρησιμοποιούνται είναι η ηλικία, το ποσό ασφάλισης και το εισόδημα,
- *AsfaliesZwhs_Fash2.arff*, όπου τα χαρακτηριστικά που χρησιμοποιούνται είναι η ηλικία και η υγεία,
- *AsfaliesZwhs_Total.arff*, όπου τα χαρακτηριστικά που χρησιμοποιούνται είναι η ηλικία, το ποσό ασφάλισης, το εισόδημα και η υγεία.

Και στις τρεις περιπτώσεις η κλάση προς αναγνώριση είναι η Ασφαλισιμότητα.

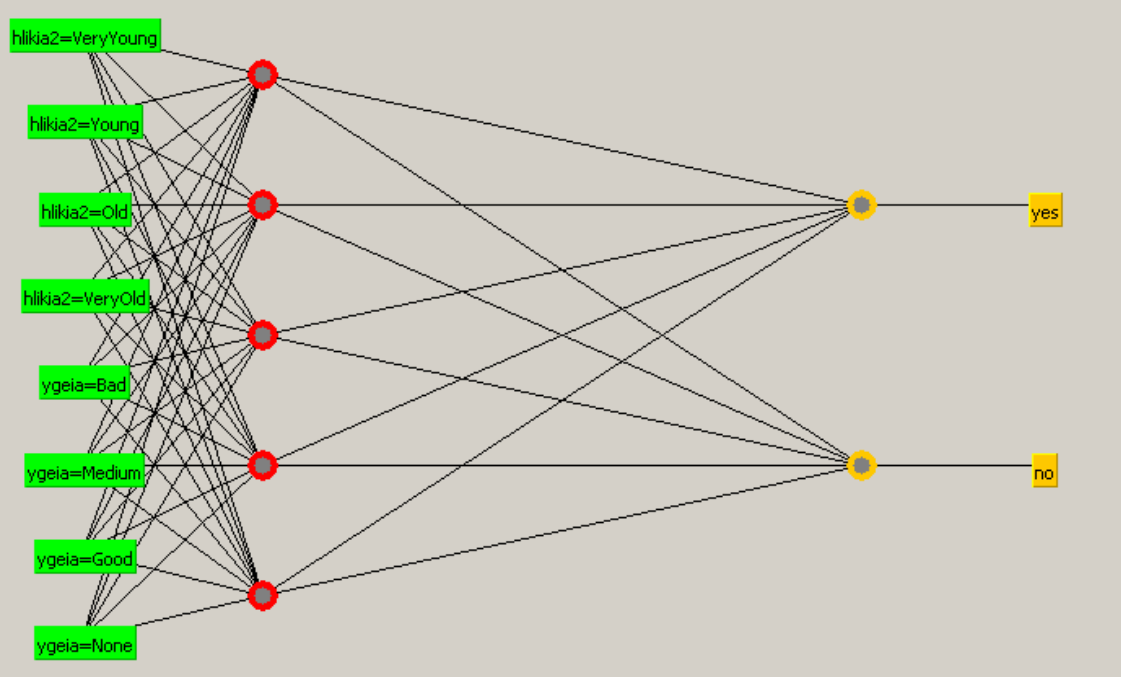
Μετά το τρέξιμο των τριών αρχείων δεδομένων από την εφαρμογή του Weka παίρνουμε τα εξής αποτελέσματα.

- Για το AsfaliesZwhs_Fash1.arff χρειάστηκαν 82.2 sec να κατασκευαστεί το νευρωνικό δίκτυο, και με απόκλιση ανά κύκλο εκπαίδευσης της τάξης του 0.1201249, είχαμε 84.2239 % ποσοστό επιτυχίας.
- Για το AsfaliesZwhs_Fash2.arff χρειάστηκαν 36.04 sec να

κατασκευαστεί το νευρωνικό δίκτυο, και με απόκλιση ανά κύκλο εκπαίδευσης της τάξης του 0.1311545, είχαμε 83.715 % ποσοστό επιτυχίας.

- Για το AsfaliesZwhs_Fash1.arff χρειάστηκαν 67.81 sec να κατασκευαστεί το νευρωνικό δίκτυο, και με απόκλιση ανά κύκλο εκπαίδευσης της τάξης του 0.1030367, είχαμε 86.514 % ποσοστό επιτυχίας.

Σχηματικά, τα νευρωνικά δίκτυα που παράγονται είναι της μορφής που νευρωνικού δικτύου του Σχήματος 6.



Σχήμα 6. Σχηματική αναπαράσταση Νευρωνικού Δικτύου για το AsfaliesZwhs_Fash2.arff.

Βλέπουμε λοιπόν, ότι παρόλο που το πρώτο σύνολο δεδομένων περιέχει λιγότερα χαρακτηριστικά από το τρίτο, χρειάζεται περισσότερο χρόνο για να δημιουργηθεί το νευρωνικό δίκτυο. Ακόμα, όπως ήταν και αναμενόμενο, το τρίτο και πιο αναλυτικό σύνολο δεδομένων, έχει μεγαλύτερο ποσοστό επιτυχίας, αλλά και μικρότερη απόκλιση ανά κύκλο εκπαίδευσης από τα άλλα δυο. Όλα όμως τα σύνολα δεδομένων, έχουν πολύ μεγάλο ποσοστό επιτυχίας, άνω του 83 %, άρα η εφαρμογή Weka μας δίδει

πολύ ακριβή αποτελέσματα, και μάλιστα σε πολύ σύντομο χρόνο.

10. Σύγκριση Μεθόδων- Σχολιασμός Μετρήσεων

Παρατηρούμε, ότι και οι τρεις εφαρμογές που χρησιμοποιήσαμε παραπάνω, ζητάνε ως αρχείο εισόδου ένα κατάλληλα διαμορφωμένο αρχείο δεδομένων, μπορούν να δεχτούν είτε λεκτικά, είτε αριθμητικά δεδομένα εισόδου, και μας δίδουν στο τέλος και ένα ποσοστό επιτυχίας του προγράμματος. Ποιο είναι όμως το πιο κατάλληλο για το πρόβλημά μας;

Πριν να συγκρίνουμε τα ποσοστά επιτυχίας κάθε μεθόδου, στον Πίνακα 11 αναλύουμε και συγκρίνουμε κάποια από τα θετικά ή/ και αρνητικά χαρακτηριστικά του κάθε προγράμματος.

FuzzyCLIPS	Mycin- Weighted	Weka
Όταν έχει να αντιμετωπίσει μεγάλα σύνολα δεδομένων αρχίζει να κάνει loops- απαιτεί περισσότερο χρόνο	Γρήγορο ακόμα και σε μεγάλα σύνολα δεδομένων	Γρήγορο και σε μεγάλα σύνολα δεδομένων
Παρέχει τη δυνατότητα να γίνει και παράλληλος υπολογισμός Ασφαλίστρων	Δεν υπολογίζει Ασφάλιστρα, κάνει μονό υπολογισμό βέλτιστων βαρών	Δεν υπολογίζει Ασφάλιστρα, αλλά παράγει Νευρωνικό Δίκτυο
Έλεγχος και ρύθμιση γίνονται με το χέρι	Αυτόματη ρύθμιση- εύρεση βέλτιστων βαρών	Δυνατότητα ρύθμισης τιμών με κατάλληλη επιλογή αλγορίθμων μάθησης, ταξινομητών, ρυθμού μάθησης, σταθεράς ορμής
Πλεονέκτημα ασάφειας	Προσέγγιση με συντελεστές βεβαιότητας	Δεν παρέχει κανόνες ασάφειας, αλλά μοντέλο με βάρη στις διαδρομές

Πίνακας 11. Σύγκριση χαρακτηριστικών τριών εφαρμογών.

Επομένως, όσον αφορά την αντιμετώπιση μεγάλων συνόλων δεδομένων, ταχύτερες είναι οι εφαρμογές Mycin- Weighted και Weka, αλλά αν αποζητάμε και άλλους παράλληλους υπολογισμούς, μόνο το FuzzyCLIPS παρέχει αυτή τη δυνατότητα.

Ακόμα, η εφαρμογή Weka παρέχει ένα πιο εύχρηστο περιβάλλον για

ρυθμίσεις παραμέτρων, και επιπλέον έχει τη δυνατότητα χρήσης της ασάφειας μέσω της δημιουργίας Νευρωνικών Δικτύων με τη χρήση βαρών.

Συγκρίνοντας, τέλος, τα ποσοστά επιτυχίας της κάθε μεθόδου παίρνουμε τον Πίνακα 12.

Ασαφείς Κανόνες-FuzzyCLIPS (%)		**Κανόνες με ΣΒ Weighted (%)**		**Νευρωνικό Δίκτυο-Weka (%)**	
Accuracy	78.86	Accuracy	80.4184	Φάση 1	84.2239
Sensitivity	91.09	Sensitivity	87.803	Φάση 2	83.715
Specificity	19.05	Specificity	40.0	Σύνολο	86.514
Precision	84.4	Precision	88.8885		

Πίνακας 12. Συνολικά αποτελέσματα μετρήσεων τριών μεθόδων.

Παρατηρούμε αρχικά, ότι το Weka μας δίνει με διαφορετικό τρόπο τα ποσοστά επιτυχίας, δίνοντας μας έτσι στην ουσία την τιμή της μετρικής Accuracy. Συγκρίνοντας τις τιμές της Accuracy λοιπόν, παρατηρούμε ότι η μέθοδος Weighted δίνει καλύτερα αποτελέσματα από τη μέθοδο του FuzzyCLIPS, ενώ η επίλυση με τη μέθοδο Weka είναι ακόμα πιο ακριβής, ακόμα και αν χρησιμοποιήσουμε σύνολο δεδομένων με λιγότερα χαρακτηριστικά (ως και το σύνολο της Φάσης 2 που χρησιμοποιεί δυο μόνο χαρακτηριστικά, δίνει ακριβέστερα αποτελέσματα από τις μεθόδους FuzzyCLIPS και Mycin-Weighted). Όμως για το νευρωνικό δίκτυο το Weka δεν δίνει αποτελέσματα για τις άλλες μετρικές.

Σημειώνεται ότι λάβαμε υπόψη τις βέλτιστες προσεγγίσεις σε κάθε μέθοδο για να κάνουμε σύγκριση ποσοστών επιτυχίας, το οποίο σημαίνει ότι πήραμε τις τιμές των μετρικών που είχαμε για τα test set στης μεθόδους FuzzyCLIPS και Weighted, το ποσοστό επιτυχίας του συνολικού συνόλου δεδομένων στην εφαρμογή Weka.

Εάν τώρα συγκρίνουμε επιμέρους τις υπόλοιπες μετρικές μεταξύ των δύο πρώτων μεθόδων, παρατηρούμε ότι το FuzzyCLIPS έχει καλύτερη απόδοση βάσει της μετρικής Sensitivity, ενώ βάσει των μετρικών Specificity και Precision, υπερτερεί το Mycin-Weighted. Καθώς λοιπόν η διαφορά στην τιμή της μετρικής Sensitivity δεν είναι και τόσο μεγάλη (μόλις της τάξης του

3%), σε γενικές γραμμές, θα λέγαμε ότι η μέθοδος Mycin-Weighted υπερτερεί της μεθόδου του FuzzyCLIPS όσον αφόρα το ποσοστό επιτυχίας.

Το συμπέρασμα λοιπόν είναι, ότι εάν επιθυμούμε αποτελέσματα με μεγαλύτερη ακρίβεια και μεγάλη ταχύτητα, μάλλον θα πρέπει να στραφούμε στην εφαρμογή Weka. Εάν όμως ζητάμε να κάνουμε και παράλληλους υπολογισμούς, η μόνη εφαρμογή που μας παρέχει αυτή τη δυνατότητα είναι το FuzzyCLIPS, δεδομένου ότι είμαστε διατεθειμένοι να θυσιάσουμε λίγη από την ταχύτητα και την ακρίβεια που επιτυγχάνεται με τη χρήση των άλλων δυο μεθόδων.

Βιβλιογραφικές Αναφορές και Ιστότοποι

[1] FuzzyCLIPS: http://ai.iit.nrc.ca/IR_public/fuzzy/FuzzyCLIPS/FuzzyCLIPSIndex.html

[2] CLIPS: http://www.ghg.net/clips/CLIPS.html

[3] Ι. Βαχάβας, Π. Κεφάλας, Φ. Κόκκορας, Η. Σακελλαρίου, «Τεχνητή Νοημοσύνη», Γ΄ Έκδοση, Β. Γκιούρδας, 2006

[4] CLIPS Basic and Advanced Programming Guide: http://clipsrules.sourceforge.net/OnlineDocs.html

[5] http://www.macs.hw.ac.uk

[6] Κ. Κόβας, «Υλοποίηση και Αξιολόγηση Μεθόδου Μετάδοσης Συντελεστών Βεβαιότητας σε Έμπειρο σύστημα», Διπλωματική Εργασία, Τμήμα Μηχ/κών Η/Υ & Πληροφορικής, Πανεπιστήμιο Πατρών, 2009.

[7] Λαγαρός Δ. Νικόλαος, Εφαρμογές των νευρωνικών δικτύων στην στατική ανάλυση και τον σχεδιασμό των κατασκευών, Διπλωματική Εργασία, Εθνικό Μετσόβιο Πολυτεχνείο, 1994.

[8] Λαγαρός Δ. Νικόλαος, Βελτιστοποίηση κατασκευών με τη χρήση εξελικτικών αλγορίθμων και νευρωνικών δικτύων, Διδακτορική Διατριβή, Εθνικό Μετσόβιο Πολυτεχνείο, 2000.

[9] Ian H. Witten; Eibe Frank (2005). "Data Mining: Practical machine learning tools and techniques, 2nd Edition". Morgan Kaufmann, San Francisco. http://www.cs.waikato.ac.nz/~ml/weka/book.html

[10] G. Holmes; A. Donkin and I.H. Witten (1994). "Weka: A machine learning workbench". Proc Second Australia and New Zealand Conference on Intelligent Information Systems, Brisbane, Australia. http://www.cs.waikato.ac.nz/~ml/publications/1994/Holmes-ANZIIS-WEKA.pdf

[11] S.R. Garner; S.J. Cunningham, G. Holmes, C.G. Nevill-Manning, and

I.H. Witten (1995). "Applying a machine learning workbench: Experience with agricultural databases". Proc Machine Learning in Practice Workshop, Machine Learning Conference, Tahoe City, CA, USA. pp. 14-21. http://www.cs.waikato.ac.nz/~ml/publications/1995/Garner95-imlc95.pdf Retrieved 2007-06-25.

[12] P. Reutemann; B. Pfahringer and E. Frank (2004). "Proper: A Toolbox for Learning from Relational Data with Propositional and Multi-Instance Learners". 17th Australian Joint Conference on Artificial Intelligence (AI2004). Springer-Verlag. http://www.cs.waikato.ac.nz/~eibe/pubs/reutemann_et_al.ps.gz

[13] Ian H. Witten; Eibe Frank, Len Trigg, Mark Hall, Geoffrey Holmes, and Sally Jo Cunningham (1999). "Weka: Practical Machine Learning Tools and Techniques with Java Implementations". Proceedings of the ICONIP/ANZIIS/ANNES'99 Workshop on Emerging Knowledge Engineering and Connectionist-Based Information Systems. pp. 192-196. http://www.cs.waikato.ac.nz/~ml/publications/1999/99IHW-EF-LT-MH-GH-SJC-Tools-Java.pdf

[14] Gregory Piatetsky-Shapiro (2005-06-28). "KDnuggets news on SIGKDD Service Award 2005". http://www.kdnuggets.com/news/2005/n13/2i.html

[15] "Overview of SIGKDD Service Award winners". 2005. http://www.acm.org/sigs/sigkdd/awards_service.php

Παράρτημα Ι

Σύνολο Δεδομένων 393 πραγματικών περιπτώσεων

0

Α/Α	ΕΙΔΟΣ ΑΣΦΑΛΕΙΑΣ	ΓΕΝΟΣ	ΗΛΙΚΙΑ	ΔΙΑΡΚΕΙΑ ΑΣΦΑΛΙΣΗΣ	ΠΟΣΟ ΑΣΦΑΛΙΣΗΣ	ΕΠΑΓΓΕΛΜΑ		ΕΤΗΣΙΟ ΕΙΣΟΔΗΜΑ	ΥΨΟΣ ΑΣΦ Π.Α.	ΥΓΕΙΑ- SCORE		ΑΣΦΑΛΙΣΙΜΟΤΗΤΑ
1	Α	Α	47	20	7337,00	28	ΔΙΚΗΓΟΡΟΣ	30000,00	29318,00			Α
2	Α	Α	28	37	1468,00	65	ΜΑΡΜΑΡΑΣ	6500,00	5870,00			Α
3	Α	Α	32	33	2935,00	112	ΔΗΜΟΣΙΟΣ ΥΠΑΛΛΗΛΟΣ	9500,00	2935,00			Α
4	Α	Α	29	30	1468,00	112	ΔΗΜΟΣΙΟΣ ΥΠΑΛΛΗΛΟΣ	9200,00	5870,00			Α
5	Α	Α	25	30	4403,00	42	ΗΛΕΚΤΡΟΛΟΓΟΣ	7950,00	7337,00			Α
6	Α	Γ	24	26	881,00	112	ΥΠΑΛΛΗΛΟΣ ΓΡΑΦΕΙΟΥ	10200,00	2935,00			Α
7	Α	Α	36	29	7337,00	34	ΕΜΠΟΡΟΣ	16100,00	14674,00			Α
8	Α	Γ	32	30	2935,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	12500,00	11739,00			Α
9	Α	Α	44	20	4403,00	115	ΦΑΝΟΠΟΙΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	6350,00	5870,00			Α
10	Α	Α	25	25	2935,00	1	ΑΓΡΟΤΗΣ	4860,00	4403,00			Α
11	Α	Α	38	25	1174,00	52	ΚΑΦΕΠΩΛΗΣ	3500,00	2348,00			Α
12	Α	Α	44	20	4403,00	70	ΜΗΧΑΝΙΚΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	6540,00	5870,00			Α
13	Α	Α	56	9	5870,00	46	ΙΑΤΡΟΣ ΑΚΤΙΝΟΛΟΓΟΣ	32140,00	29348,00			Μ
14	Α	Γ	35	25	3620,00	34	ΕΜΠΟΡΟΣ	16530,00	5870,00			Α
15	Α	Α	36	20	8345,00	35	ΕΠΙΧΕΙΡΗΜΑΤΙΑΣ	23540,00	22658,00			Α
16	Α	Γ	52	10	2935,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	16320,00	16300,00			Μ
17	Α	Α	44	25	25000,00	46	ΙΑΤΡΟΣ ΧΕΙΡΟΥΡΓΟΣ	43250,00	38000,00			Α
18	Α	Γ	28	37	20000,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	25000,00	25000,00			Μ
19	Α	Γ	38	20	15000,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	14500,00	12000,00			Α
20	Α	Γ	42	23	47000,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	62000,00	20000,00	Good	0,670	Α
21	Α	Α	32	25	10250,00	52	ΚΑΦΕΠΩΛΗΣ	5840,00	4210,00			Α
22	Α	Α	38	20	8300,00	35	ΕΠΙΧΕΙΡΗΜΑΤΙΑΣ	5620,00	2350,00			Α
23	Α	Α	49	15	1230,00	59	ΛΑΤΟΜΟΣ	10540,00	2030,00			Α
24	Α	Γ	45	12	14520,00	34	ΕΜΠΟΡΟΣ	8650,00	13200,00			Μ
25	Α	Γ	60	5	30100,00	55	ΚΟΜΜΩΤΗΣ	67500,00	15000,00	Medium	0,614	Α
26	Α	Γ	42	15	12300,00	114	ΥΦΑΝΤΟΥΡΓΟΣ	7520,00	5620,00			Α
27	Α	Α	30	35	22300,00	76	ΝΑΥΤΙΚΟΣ	35000,00	30000,00			Μ
28	Α	Α	40	25	7410,00	90	ΠΡΑΤΗΡΙΟΥΧΟΣ ΥΓΡ ΚΑΥΣ	5800,00	7000,00			Μ
29	Α	Α	36	29	1620,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	5610,00	1650,00			Α
30	Α	Α	44	20	1874,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	5230,00	1658,00			Α
31	Α	Α	35	30	1620,00	14	ΒΙΟΤΕΧΝΗΣ	4540,00	2030,00			Α
32	Α	Α	39	20	43800,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	58410,00	21300,00			Α
33	Α	Α	32	30	1468,00	1	ΑΓΡΟΤΗΣ	5540,00	1750,00			Α
34	Α	Γ	57	12	10000,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	14500,00	3410,00			Α
35	Α	Α	43	22	3200,00	111	ΥΔΡΑΥΛΙΚΟΣ	6740,00	10000,00			Μ

36	Α	Α	35	25	45000,00	88	ΠΙΛΟΤΟΣ	37000,00	37000,00			Μ
37	Α	Α	26	35	78000,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	85400,00	27000,00	Medium	0,600	Α
38	Α	Α	46	15	2350,00	90	ΠΡΑΤΗΡΙΟΥΧΟΣ ΥΓΡ ΚΑΥΣ	5680,00	5500,00			Α
39	Α	Γ	34	20	1350,00	73	ΜΟΥΣΙΚΟΣ	8640,00	7000,00			Μ
40	Α	Α	38	27	23000,00	28	ΔΙΚΗΓΟΡΟΣ	37590,00	17000,00			Α
41	Α	Α	43	22	1468,00	94	ΙΔ ΥΠΑΛΛΗΛΟΣ	10240,00	2350,00			Α
42	Α	Α	50	15	2100,00	35	ΕΠΙΧΕΙΡΗΜΑΤΙΑΣ	6030,00	3100,00			Α
43	Α	Γ	45	15	2935,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	16320,00	16300,00			Μ
44	Α	Α	47	15	45200,00	99	ΣΥΜΒΟΛΑΙΟΓΡΑΦΟΣ	25800,00	20000,00	Medium	0,524	Μ
45	Α	Α	22	30	50000,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	80000,00	30000,00	Good	0,930	Α
46	Α	Α	51	20	4200,00	60	ΛΑΧΕΙΟΠΩΛΗΣ	5200,00	5000,00			Α
47	Α	Α	30	30	68000,00	105	ΤΟΠΟΓΡΑΦΟΣ	68000,00	23000,00	Good	0,840	Α
48	Α	Α	32	20	8300,00	85	ΕΡΓΑΤΗΣ ΟΙΚΟΔΟΜΗΣ	13560,00	13000,00			Α
49	Α	Γ	57	8	45800,00	84	ΟΔΟΝΤΟΤΕΧΝΙΤΗΣ	53100,00	19580,00	Medium	0,570	Α
50	Α	Γ	53	12	5300,00	34	ΕΜΠΟΡΟΣ	8530,00	8000,00			Α
51	Α	Α	55	10	33000,00	43	ΗΛΕΚΤΡ- ΜΗΧΑΝΟΛ	30000,00	17420,00	Good	0,730	Α
52	Α	Γ	37	28	881,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	9780,00	1258,00			Α
53	Α	Α	24	20	1000,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	15500,00	2500,00			Α
54	Α	Α	32	15	1500,00	99	ΣΥΜΒΟΛΑΙΟΓΡΑΦΟΣ	16200,00	10000,00			Μ
55	Α	Α	39	10	1700,00	1	ΑΓΡΟΤΗΣ	15100,00	2000,00			Α
56	Α	Γ	25	25	5000,00	34	ΕΜΠΟΡΟΣ	4000,00	4000,00			Α
57	Α	Α	27	30	2300,00	70	ΜΗΧΑΝΙΚΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	4300,00	2500,00			Α
58	Α	Α	38	22	5900,00	52	ΚΑΦΕΠΩΛΗΣ	3550,00	3000,00			Α
59	Α	Α	40	25	4300,00	115	ΦΑΝΟΠΟΙΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	4500,00	2310,00			Α
60	Α	Α	35	20	6200,00	65	ΜΑΡΜΑΡΑΣ	4500,00	3580,00			Α
61	Α	Γ	37	17	10000,00	34	ΕΜΠΟΡΟΣ	4100,00	3000,00			Α
62	Α	Α	45	20	1300,00	59	ΛΑΤΟΜΟΣ	3580,00	5000,00			Μ
63	Α	Α	50	12	1500,00	70	ΜΗΧΑΝΙΚΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	4100,00	3500,00			Α
64	Α	Α	48	12	1100,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	3900,00	3900,00			Α
65	Α	Α	42	20	1300,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	15000,00	10000,00			Μ
66	Α	Γ	47	13	1100,00	35	ΕΠΙΧΕΙΡΗΜΑΤΙΑΣ	17300,00	3300,00			Α
67	Α	Α	50	15	1700,00	34	ΕΜΠΟΡΟΣ	16100,00	4000,00			Α
68	Α	Γ	42	20	5000,00	55	ΚΟΜΜΩΤΗΣ	3500,00	3200,00			Α
69	Α	Α	48	12	3800,00	1	ΑΓΡΟΤΗΣ	4300,00	4000,00			Α
70	Α	Α	55	5	7300,00	37	ΕΣΤΙΑΤΟΡΑΣ	4500,00	1500,00			Α
71	Α	Α	48	17	11000,00	47	ΙΕΡΕΑΣ	4200,00	3000,00			Α
72	Α	Γ	42	23	8520,00	50	ΖΑΧΑΡΟΠΛΑΣΤΗΣ	4380,00	4300,00			Α
73	Α	Α	20	20	1500,00	1	ΑΓΡΟΤΗΣ	4450,00	1000,00			Α
74	Α	Α	16	30	1300,00	96	ΣΕΡΒΙΤΟΡΟΣ	3500,00	1200,00			Α
75	Α	Α	22	28	3400,00	117	ΦΟΙΤΗΤΗΣ	4310,00	2030,00			Α
76	Α	Γ	21	10	1800,00	58	ΚΤΙΣΤΗΣ	3510,00	3000,00			Α
77	Α	Α	20	35	1000,00	10	ΑΣΦΑΛΙΣΤΗΣ	4150,00	3200,00			Α
78	Α	Α	21	44	6300,00	30	ΕΙΣΠΡΑΚΤΟΡΑΣ	4600,00	3200,00			Α
79	Α	Α	18	30	3050,00	85	ΟΙΚΟΔΟΜΟΣ	5940,00	5900,00			Α
80	Α	Α	19	11	1800,00	111	ΥΔΡΑΥΛΙΚΟΣ	7880,00	6000,00			Α
81	Α	Α	18	20	1500,00	1	ΑΓΡΟΤΗΣ	6700,00	6200,00			Μ
82	Α	Α	16	24	1300,00	117	ΦΟΙΤΗΤΗΣ	3800,00	1300,00			Α
83	Α	Α	19	21	1600,00	6	ΑΠΟΘΗΚΑΡΙΟΣ	12100,00	8000,00			Μ
84	Α	Α	22	15	2000,00	36	ΕΠΙΠΛΟΠΟΙΟΣ	8230,00	5400,00			Α
85	Α	Α	22	18	3200,00	96	ΣΕΡΒΙΤΟΡΟΣ	12310,00	5000,00			Α

86	Α	Α	21	30	7000,00	50	ΖΑΧΑΡΟΠΛΑΣΤΗΣ	15530,00	10000,00			Α
87	Α	Γ	22	33	6300,00	96	ΣΕΡΒΙΤΟΡΟΣ	15300,00	9500,00			Α
88	Α	Α	19	26	1300,00	65	ΜΑΡΜΑΡΑΣ	6340,00	6000,00			Μ
89	Α	Α	20	35	1800,00	36	ΕΠΙΠΛΟΠΟΙΟΣ	10200,00	7000,00			Α
90	Α	Α	21	42	6310,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	11430,00	8300,00			Α
91	Α	Γ	22	33	1700,00	3	ΑΙΣΘΗΤΙΚΟΣ	15100,00	3000,00			Α
92	Α	Γ	22	33	3100,00	104	ΤΗΛΕΦΩΝΗΤΡΙΑ	15380,00	3100,00			Α
93	Α	Α	22	43	6850,00	74	ΜΠΑΡΜΑΝ	16020,00	10000,00			Α
94	Α	Α	16	40	2000,00	64	ΜΑΘΗΤΗΣ	4810,00	1000,00			Α
95	Α	Γ	40	20	46000,00	105	ΤΟΠΟΓΡΑΦΟΣ	58000,00	32000,00	Medium	0,610	Α
96	Α	Α	39	26	45300,00	24	ΔΗΜΟΣΙΟΓΡΑΦΟΣ	51200,00	17000,00	Bad	0,400	Μ
97	Α	Γ	55	10	32000,00	40	ΖΩΓΡΑΦΟΣ	100000,00	50000,00	Medium	0,600	Α
98	Α	Α	52	10	31200,00	37	ΕΣΤΙΑΤΟΡΑΣ	63000,00	20000,00	Bad	0,390	Μ
99	Α	Γ	50	15	30100,00	105	ΤΟΠΟΓΡΑΦΟΣ	47000,00	10000,00	Medium	0,550	Α
100	Α	Α	56	5	24100,00	37	ΕΣΤΙΑΤΟΡΑΣ	42000,00	15000,00	Medium	0,480	Μ
101	Α	Α	58	7	26400,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	37520,00	9320,00	Medium	0,470	Μ
102	Α	Α	55	10	32100,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	48520,00	20000,00	Medium	0,530	Α
103	Α	Α	46	19	30100,00	34	ΕΜΠΟΡΟΣ	45300,00	11000,00	Medium	0,480	Μ
104	Α	Α	50	8	45320,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	72100,00	35000,00	Medium	0,500	Μ
105	Α	Α	46	10	48540,00	28	ΔΙΚΗΓΟΡΟΣ	54800,00	20000,00	Medium	0,446	Μ
106	Α	Γ	50	10	45610,00	26	ΔΙΑΚΟΣΜΗΤΗΣ	63410,00	12000,00	Medium	0,430	Μ
107	Α	Α	47	13	45400,00	41	ΗΘΟΠΟΙΟΣ	52720,00	14500,00	Bad	0,302	Μ
108	Α	Α	38	20	60500,00	38	ΕΦΟΠΛΙΣΤΗΣ	92300,00	27400,00	Medium	0,626	Α
109	Α	Γ	32	25	62100,00	26	ΔΙΑΚΟΣΜΗΤΗΣ	63870,00	17800,00	Medium	0,614	Α
110	Α	Α	60	5	50000,00	105	ΤΟΠΟΓΡΑΦΟΣ	52000,00	21000,00	Bad	0,336	Μ
111	Α	Α	56	9	45300,00	97	ΣΚΗΝΟΘΕΤΗΣ	65240,00	15000,00	Bad	0,392	Μ
112	Α	Α	58	7	50000,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	57300,00	11000,00	Medium	0,426	Μ
113	Α	Α	60	5	48000,00	38	ΕΦΟΠΛΙΣΤΗΣ	63400,00	30000,00	Bad	0,388	Μ
114	Α	Γ	56	5	46100,00	56	ΚΟΣΜΗΜΑΤΟΠΩΛΗΣ	72500,00	24000,00	Bad	0,342	Μ
115	Α	Α	59	5	45100,00	34	ΕΜΠΟΡΟΣ	59540,00	12000,00	Medium	0,440	Μ
116	Α	Α	45	15	45130,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	50000,00	22300,00	Medium	0,650	Α
117	Α	Α	44	10	49000,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	64100,00	15000,00	Good	0,790	Α
118	Α	Α	34	16	48000,00	24	ΔΗΜΟΣΙΟΓΡΑΦΟΣ	53000,00	12000,00	Good	0,670	Α
119	Α	Γ	23	30	46000,00	72	ΜΟΝΤΕΛΟ	74520,00	20000,00	Good	0,660	Α
120	Α	Α	32	20	48630,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	62540,00	14200,00	Bad	0,200	Μ
121	Α	Α	39	12	50000,00	13	ΒΙΟΜΗΧΑΝΟΣ	70000,00	17000,00	Medium	0,520	Α
122	Α	Γ	40	25	45510,00	41	ΙΑΤΡΟΣ	57300,00	23000,00	Medium	0,410	Μ
123	Α	Α	32	33	2935,00	112	ΔΗΜΟΣΙΟΣ ΥΠΑΛΛΗΛΟΣ	9500,00	12450,00			Μ
124	Α	Α	32	30	1680,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	9500,00	3500,00			Μ
125	Α	Α	29	20	6800,00	112	ΔΗΜΟΣΙΟΣ ΥΠΑΛΛΗΛΟΣ	9200,00	20000,00			Μ
126	Α	Α	42	15	4403,00	29	ΔΥΤΗΣ	14500,00	8000,00			Μ
127	Α	Γ	50	10	1600,00	2	ΑΘΛΗΤΗΣ	5200,00	3500,00			Μ
128	Α	Α	48	8	12540,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	11254,00	32000,00			Μ
129	Α	Α	42	23	1200,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	14510,00	4900,00			Μ
130	Α	Α	25	30	3100,00	4	ΑΝΘΡΑΚΩΡΥΧΟΣ	15100,00	12000,00			Μ
131	Α	Α	30	35	1500,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	22450,00	22000,00			Μ
132	Α	Γ	30	25	6100,00	104	ΤΗΛΕΦΩΝΗΤΡΙΑ	15100,00	18400,00			Μ
133	Α	Α	55	10	4500,00	76	ΝΑΥΤΙΚΟΣ	35000,00	12000,00			Μ
134	Α	Α	42	10	1600,00	1	ΑΓΡΟΤΗΣ	15030,00	6500,00			Μ
135	Α	Α	48	8	15100,00	88	ΠΙΛΟΤΟΣ	31240,00	25000,00			Μ

136	I	A	44	-	1468,00	94	ΠΩΛΗΤΗΣ	9250,00	2202,00			A
137	I	A	44	-	1468,00	36	ΕΠΙΠΛΟΠΟΙΟΣ	9780,00	2202,00			A
138	I	A	51	-	2055,00	59	ΕΠΙΒΛ ΛΑΤΟΜΕΙΟΥ	5390,00	2935,00			A
139	I	A	49	-	1028,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	11530,00	1468,00			A
140	I	A	36	-	1468,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	4510,00	1468,00			A
141	I	A	44	-	1174,00	61	ΛΙΜΕΝΟΦΥΛΑΚΑΣ	4850,00	1468,00			A
142	I	A	26	-	2935,00	67	ΛΕΥΚΟΣΙΔΗΡΟΥΡΓΟΣ	5360,00	2348,00			A
143	I	A	35	-	1468,00	67	ΜΕΤΑΛΛΟΥΡΓΟΣ	6310,00	1468,00			A
144	I	A	42	-	2935,00	57	ΚΤΗΝΟΤΡΟΦΟΣ	6520,00	6000,00			A
145	I	A	41	-	1468,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	15630,00	14674,00			M
146	I	A	49	-	881,00	115	ΦΑΝΟΠΟΙΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	5320,00	1468,00			A
147	I	A	29	-	1174,00	20	ΓΟΥΝΟΠΟΙΟΣ	4630,00	1468,00			A
148	I	Γ	42	-	881,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	4680,00	1468,00			A
149	I	Γ	23	-	10000,00	86	ΟΙΚΙΑΚΑ	12500,00	6000,00			A
150	I	A	52	-	1420,00	54	ΚΗΠΟΥΡΟΣ	5120,00	2530,00			A
151	I	A	57	-	35000,00	76	ΝΑΥΤΙΚΟΣ	30000,00	15000,00			M
152	I	A	52	-	32140,00	73	ΜΟΥΣΙΚΟΣ	36540,00	15620,00	Good	0,670	A
153	I	A	50	-	2560,00	45	ΘΥΡΩΡΟΣ	9850,00	2910,00			A
154	I	Γ	47	-	2160,00	77	ΝΗΠΙΑΓΩΓΟΣ	13420,00	3540,00			A
155	I	Γ	42	-	881,00	45	ΘΥΡΩΡΟΣ	7520,00	3560,00			M
156	I	A	35	-	20000,00	29	ΔΥΤΗΣ	16580,00	16500,00			M
157	I	A	47	-	887,00	85	ΟΙΚΟΔΟΜΟΣ	4680,00	887,00			A
158	I	A	56	-	37540,00	20	ΓΟΥΝΟΠΟΙΟΣ	54210,00	50000,00	Medium	0,528	A
159	I	A	28	-	974,00	7	ΑΡΤΟΠΟΙΟΣ	8560,00	3580,00			A
160	I	A	28	-	5000,00	115	ΦΑΝΟΠΟΙΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	4630,00	1468,00			A
161	I	A	19	-	1258,00	67	ΜΕΤΑΛΛΟΥΡΓΟΣ	7840,00	2130,00			A
162	I	A	36	-	3210,00	21	ΓΡΑΦΙΣΤΑΣ	16840,00	14320,00			M
163	I	Γ	32	-	15000,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	23540,00	20000,00			A
164	I	A	40	-	950,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	10250,00	3100,00			A
165	I	Γ	51	-	3540,00	34	ΕΜΠΟΡΟΣ	4300,00	2450,00			A
166	I	A	46	-	5320,00	39	ΖΑΧΑΡΟΠΛΑΣΤΗΣ	17840,00	12560,00			A
167	I	A	33	-	7420,00	57	ΚΤΗΝΟΤΡΟΦΟΣ	6310,00	2500,00			A
168	I	Γ	46	-	5230,00	7	ΑΡΤΟΠΟΙΟΣ	11250,00	2310,00			A
169	I	A	57	-	32000,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	43000,00	32000,00	Medium	0,452	M
170	I	Γ	46	-	42100,00	40	ΖΩΓΡΑΦΟΣ	98000,00	30000,00	Good	0,670	A
171	I	Γ	40	-	5420,00	116	ΦΑΡΜΑΚΟΠΟΙΟΣ	4680,00	1468,00			A
172	I	A	27	-	100000,00	13	ΒΙΟΜΗΧΑΝΟΣ	100000,00	45000,00	Good	0,890	A
173	I	A	49	-	2510,00	54	ΚΗΠΟΥΡΟΣ	7620,00	5420,00			A
174	I	A	39	-	62100,00	84	ΟΔΟΝΤΟΤΕΧΝΙΤΗΣ	84120,00	25410,00	Medium	0,564	A
175	I	A	32	-	15000,00	93	ΠΥΡΟΣΒΕΣΤΗΣ	17520,00	10000,00			A
176	I	A	35	-	2510,00	83	ΟΔΟΚΑΘΑΡΙΣΤΗΣ	4680,00	1587,00			A
177	I	A	50	-	2350,00	47	ΙΕΡΕΑΣ	17450,00	2350,00			A
178	I	Γ	55	-	43000,00	105	ΤΟΠΟΓΡΑΦΟΣ	48000,00	30000,00	Medium	0,470	M
179	I	A	57	-	49050,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	65000,00	12000,00	Bad	0,368	M
180	I	A	42	-	46840,00	105	ΤΟΠΟΓΡΑΦΟΣ	57410,00	31000,00	Good	0,660	A
181	I	A	40	-	45120,00	24	ΔΗΜΟΣΙΟΓΡΑΦΟΣ	48520,00	16550,00	Medium	0,530	A
182	I	A	58	-	52000,00	38	ΕΦΟΠΛΙΣΤΗΣ	73600,00	25000,00	Medium	0,414	M
183	I	A	50	-	34200,00	56	ΚΟΣΜΗΜΑΤΟΠΩΛΗΣ	65000,00	17000,00	Medium	0,600	A
184	I	Γ	49	-	31230,00	102	ΣΧΕΔΙΑΣΤΡΙΑ	46410,00	8600,00	Good	0,670	A
185	I	A	57	-	25400,00	37	ΕΣΤΙΑΤΟΡΑΣ	35000,00	13650,00	Bad	0,260	M

186	I	A	58	-	27340,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	43500,00	10000,00	Medium	0,410	M
187	I	Γ	46	-	47510,00	34	ΕΜΠΟΡΟΣ	75100,00	21000,00	Medium	0,470	M
188	I	A	57	-	31000,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	44300,00	5630,00	Medium	0,454	M
189	I	A	41	-	45100,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	75100,00	25000,00	Good	0,860	A
190	I	A	55	-	45270,00	97	ΣΚΗΝΟΘΕΤΗΣ	78400,00	20000,00	Medium	0,516	M
191	I	Γ	61	-	45630,00	26	ΔΙΑΚΟΣΜΗΤΗΣ	64100,00	13200,00	Medium	0,518	M
192	I	A	35	-	42130,00	40	ΖΩΓΡΑΦΟΣ	62870,00	15000,00			A
193	I	A	54	-	47300,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	82400,00	21000,00	Medium	0,418	M
194	I	Γ	53	-	35100,00	41	ΗΘΟΠΟΙΟΣ	103000,00	42000,00	Medium	0,530	A
195	I	Γ	61	-	52130,00	105	ΤΟΠΟΓΡΑΦΟΣ	55000,00	10000,00	Medium	0,496	M
196	I	A	35	-	46380,00	24	ΔΗΜΟΣΙΟΓΡΑΦΟΣ	45000,00	10000,00	Good	0,930	A
197	I	A	57	-	47800,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	48100,00	10000,00	Medium	0,516	M
198	I	A	39	-	45100,00	41	ΙΑΤΡΟΣ	57610,00	20000,00	Good	0,800	A
199	I	A	40	-	51000,00	28	ΔΙΚΗΓΟΡΟΣ	51000,00	10000,00	Good	0,660	A
200	I	A	58	-	48300,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	61300,00	8000,00	Medium	0,522	A
201	I	A	41	-	44300,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	47640,00	13000,00			A
202	I	A	35	-	28650,00	38	ΕΦΟΠΛΙΣΤΗΣ	46000,00	11000,00			A
203	I	A	32	-	45100,00	24	ΔΗΜΟΣΙΟΓΡΑΦΟΣ	48000,00	10000,00	Good	0,860	A
204	I	A	40	-	50600,00	13	ΒΙΟΜΗΧΑΝΟΣ	81450,00	15000,00	Medium	0,600	A
205	I	A	27	-	47100,00	72	ΜΟΝΤΕΛΟ	82100,00	20000,00	Good	1,000	A
206	I	Γ	41	-	45320,00	27	ΔΙΑΦΗΜΙΣΤΗΣ	51000,00	18000,00	Good	0,730	A
207	I	A	42	-	5000,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	47300,00	10000,00			A
208	I	A	47	-	5850,00	33	ΕΛΑΙΟΧΡΩΜΑΤΙΣΤΗΣ	45100,00	6000,00			A
209	I	A	51	-	7000,00	34	ΕΜΠΟΡΟΣ	45800,00	12000,00			A
210	I	Γ	54	-	10000,00	46	ΙΑΤΡΟΣ ΧΕΙΡΟΥΡΓΟΣ	51000,00	20000,00			A
211	I	A	45	-	17000,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	49000,00	32000,00			M
212	I	A	51	-	13200,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	53300,00	15000,00			A
213	I	A	50	-	1630,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	45300,00	4500,00			A
214	I	A	44	-	1300,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	49130,00	4000,00			A
215	I	Γ	60	-	2800,00	14	ΒΙΟΤΕΧΝΗΣ	5100,00	6000,00			M
216	I	Γ	61	-	4300,00	50	ΚΑΘΑΡΙΣΤΡΙΑ	4120,00	4000,00			A
217	I	A	60	-	3900,00	1	ΑΓΡΟΤΗΣ	3900,00	3500,00			A
218	I	Γ	56	-	5000,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	11120,00	5000,00			A
219	I	A	57	-	1800,00	42	ΗΛΕΚΤΡΟΛΟΓΟΣ	10300,00	10000,00			M
220	I	Γ	57	-	5130,00	78	ΝΟΣΟΚΟΜΟΣ	8350,00	8000,00			A
221	I	A	57	-	5300,00	31	ΕΚΔΟΤΗΣ	62100,00	15000,00			A
222	I	A	58	-	4680,00	84	ΟΔΟΝΤΟΤΕΧΝΙΤΗΣ	53400,00	12000,00			A
223	I	A	57	-	5430,00	105	ΤΟΠΟΓΡΑΦΟΣ	51000,00	12300,00			A
224	I	A	59	-	15000,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	18500,00	9800,00			A
225	I	A	56	-	17300,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	16100,00	15000,00			A
226	I	A	58	-	19500,00	99	ΣΥΜΒΟΛΑΙΟΓΡΑΦΟΣ	48100,00	10000,00			A
227	I	A	57	-	15000,00	28	ΔΙΚΗΓΟΡΟΣ	49130,00	12100,00			A
228	I	A	59	-	12350,00	99	ΣΥΜΒΟΛΑΙΟΓΡΑΦΟΣ	52100,00	12000,00			A
229	I	A	60	-	16300,00	46	ΙΑΤΡΟΣ	60300,00	15000,00			A
230	I	A	60	-	18000,00	41	ΗΘΟΠΟΙΟΣ	47000,00	18000,00			A
231	I	Γ	62	-	46100,00	105	ΤΟΠΟΓΡΑΦΟΣ	37800,00	6000,00	Medium	0,420	M
232	I	A	58	-	1300,00	94	ΠΩΛΗΤΗΣ	4100,00	2000,00			A
233	I	A	60	-	900,00	63	ΜΑΓΕΙΡΑΣ	12800,00	2100,00			A
234	I	A	60	-	1100,00	54	ΚΗΠΟΥΡΟΣ	10300,00	3000,00			A
235	I	Γ	56	-	1500,00	60	ΛΑΧΕΙΟΠΩΛΗΣ	7830,00	4000,00			A

236	I	Γ	60	-	1680,00	1	ΑΓΡΟΤΗΣ	6500,00	2530,00			A
237	I	A	22	-	21000,00	58	ΚΤΙΣΤΗΣ	45600,00	15000,00			A
238	I	A	21	-	23100,00	20	ΓΟΥΝΟΠΟΙΟΣ	47100,00	35000,00			A
239	I	A	22	-	5000,00	108	ΤΥΠΟΓΡΑΦΟΣ	48100,00	20000,00			A
240	I	A	21	-	5300,00	40	ΖΩΓΡΑΦΟΣ	45130,00	18650,00			A
241	I	A	20	-	4800,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	4900,00	4000,00			A
242	I	A	22	-	4100,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	52100,00	12000,00			A
243	I	A	21	-	5900,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	51800,00	16000,00			A
244	I	A	19	-	8000,00	14	ΒΙΟΤΕΧΝΗΣ	47300,00	24000,00			A
245	I	A	21	-	10000,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	45800,00	25000,00			A
246	I	A	19	-	15000,00	41	ΗΘΟΠΟΙΟΣ	50700,00	20000,00			A
247	I	Γ	16	-	48000,00	86	ΟΙΚΙΑΚΑ	4500,00	25000,00	Good	0,940	M
248	I	Γ	18	-	47100,00	73	ΜΟΥΣΙΚΟΣ	13100,00	10000,00	Good	0,800	A
249	I	A	19	-	50000,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	14000,00	8000,00	Good	0,930	A
250	I	A	17	-	48800,00	85	ΟΙΚΟΔΟΜΟΣ	12000,00	5600,00	Good	0,860	A
251	I	Γ	20	-	47300,00	1	ΑΓΡΟΤΗΣ	13900,00	10000,00	Good	0,730	A
252	I	A	22	-	51300,00	76	ΝΑΥΤΙΚΟΣ	38000,00	12420,00	Good	0,870	M
253	I	A	21	-	60300,00	37	ΕΣΤΙΑΤΟΡΑΣ	40000,00	10000,00	Good	0,686	A
254	I	A	38	-	40000,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	13100,00	12450,00			M
255	I	A	25	-	5000,00	28	ΔΙΚΗΓΟΡΟΣ	47000,00	15000,00			A
256	I	A	31	-	4500,00	28	ΔΙΚΗΓΟΡΟΣ	45100,00	10000,00			A
257	I	A	30	-	3500,00	37	ΕΣΤΙΑΤΟΡΑΣ	48300,00	10300,00			A
258	I	Γ	30	-	5500,00	71	ΜΟΔΙΣΤΡΑ	49900,00	10000,00			A
259	I	A	40	-	5300,00	46	ΙΑΤΡΟΣ	48400,00	15000,00			A
260	I	A	39	-	4000,00	57	ΚΤΗΝΟΤΡΟΦΟΣ	45800,00	12000,00			A
261	I	A	25	-	15000,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	45900,00	30000,00			A
262	I	A	27	-	8000,00	14	ΒΙΟΤΕΧΝΗΣ	48000,00	8000,00			A
263	I	A	32	-	13300,00	26	ΔΙΑΚΟΣΜΗΤΗΣ	48300,00	20000,00			A
264	I	A	30	-	18000,00	72	ΜΟΝΤΕΛΟ	50120,00	20000,00			A
265	I	A	38	-	15000,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	51100,00	15000,00			A
266	I	A	36	-	13100,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	49830,00	17000,00			A
267	I	A	38	-	1600,00	46	ΙΑΤΡΟΣ	47380,00	5000,00			A
268	I	Γ	39	-	1100,00	26	ΔΙΑΚΟΣΜΗΤΗΣ	46830,00	4000,00			A
269	I	A	29	-	1650,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	47910,00	4500,00			A
270	I	A	31	-	1500,00	28	ΔΙΚΗΓΟΡΟΣ	46710,00	4000,00			A
271	I	A	52	-	48300,00	26	ΔΙΑΚΟΣΜΗΤΗΣ	35000,00	20000,00	Bad	0,368	M
272	I	A	60	-	32100,00	24	ΔΗΜΟΣΙΟΓΡΑΦΟΣ	47070,00	15000,00	Medium	0,492	M
273	I	A	58	-	5000,00	47	ΙΕΡΕΑΣ	17300,00	10000,00			A
274	I	Γ	21	-	4500,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	17000,00	9000,00			A
275	I	A	22	-	1300,00	43	ΗΛΕΚΤΡΟΛΟΓΟΣ	19100,00	4500,00			A
276	I	Γ	37	-	45300,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	27300,00	17000,00	Good	0,670	A
277	I	A	28	-	45100,00	36	ΕΠΙΠΛΟΠΟΙΟΣ	19100,00	10000,00	Good	0,860	A
278	I	A	32	-	1300,00	118	ΦΥΛΑΚΑΣ	4300,00	4500,00			M
279	I	A	30	-	1700,00	111	ΥΔΡΑΥΛΙΚΟΣ	4480,00	5000,00			M
280	M	A	32	28	1468,00	43	ΗΛΕΚΤΡ- ΜΗΧΑΝΟΛ	17520,00	2202,00			A
281	M	A	33	27	1468,00	37	ΕΣΤΙΑΤΟΡΑΣ	5500,00	1468,00			A
282	M	A	47	18	887,00	85	ΟΙΚΟΔΟΜΟΣ	6680,00	887,00			A
283	M	A	50	15	1174,00	125	ΨΗΤΟΠΩΛΗΣ	7300,00	1468,00			A
284	M	A	34	31	1468,00	70	ΜΗΧΑΝΙΚΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	4820,00	2935,00			A
285	M	A	34	21	1468,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	4600,00	1468,00			A

286	M	A	37	28	2935,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	6840,00	2935,00			A
287	M	Γ	22	28	1468,00	14	ΒΙΟΤΕΧΝΗΣ	4860,00	1468,00			A
288	M	A	37	28	887,00	61	ΛΙΜΕΝΟΦΥΛΑΚΑΣ	8130,00	881,00			A
289	M	A	30	30	1468,00	1	ΑΓΡΟΤΗΣ	4970,00	1761,00			A
290	M	Γ	38	20	881,00	71	ΜΟΔΙΣΤΡΑ	4530,00	4500,00			M
291	M	Γ	38	20	1420,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	14500,00	5000,00			A
292	M	A	32	20	1200,00	106	ΤΟΡΝΑΔΟΡΟΣ	5840,00	4800,00			A
293	M	Γ	53	12	1183,00	86	ΟΙΚΙΑΚΑ	4200,00	2800,00			A
294	M	A	36	20	12360,00	47	ΙΕΡΕΑΣ	16320,00	15000,00			A
295	M	Γ	38	22	95000,00	13	ΒΙΟΜΗΧΑΝΟΣ	100000,00	41200,00	Good	0,694	A
296	M	A	48	14	36840,00	41	ΗΘΟΠΟΙΟΣ	53210,00	30000,00	Good	0,660	A
297	M	Γ	25	20	1850,00	94	ΠΩΛΗΤΗΣ	8620,00	7000,00			A
298	M	A	36	10	4350,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	17650,00	14230,00			M
299	M	A	32	20	8450,00	1	ΑΓΡΟΤΗΣ	5540,00	2750,00			A
300	M	A	36	29	4334,00	81	ΕΠΑΓΓΕΛΜΑΤΙΑΣ ΟΔΗΓΟΣ	14250,00	1350,00			A
301	M	A	43	12	25310,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	43520,00	27850,00			A
302	M	Γ	40	10	62700,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	58000,00	25000,00	Medium	0,642	A
303	M	A	46	19	5360,00	47	ΙΕΡΕΑΣ	13520,00	10000,00			A
304	M	Γ	55	5	8500,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	12500,00	3110,00			A
305	M	A	29	30	35000,00	76	ΝΑΥΤΙΚΟΣ	39450,00	25000,00			M
306	M	Γ	37	13	1250,00	112	ΔΗΜΟΣΙΟΣ ΥΠΑΛΛΗΛΟΣ	14620,00	14500,00			M
307	M	Γ	48	17	23120,00	28	ΔΙΚΗΓΟΡΟΣ	38530,00	20000,00			A
308	M	A	53	12	2350,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	6840,00	6800,00			A
309	M	A	44	21	1860,00	87	ΟΠΩΡΟΠΩΛΗΣ	5210,00	5000,00			A
310	M	A	38	20	17560,00	126	ΨΥΚΤΙΚΟΣ	27510,00	25000,00			A
311	M	A	48	15	83120,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	98200,00	32000,00	Bad	0,368	M
312	M	A	34	15	7530,00	63	ΜΑΓΕΙΡΑΣ	10520,00	9500,00			A
313	M	Γ	58	5	61200,00	97	ΣΚΗΝΟΘΕΤΗΣ	61200,00	35000,00	Bad	0,316	M
314	M	A	39	20	3450,00	5	ΑΝΘΟΠΩΛΗΣ	8540,00	8500,00			A
315	M	Γ	46	10	10320,00	34	ΕΜΠΟΡΟΣ	11250,00	5460,00			A
316	M	A	49	16	20000,00	28	ΔΙΚΑΣΤΗΣ	32140,00	30000,00			A
317	M	A	43	12	3200,00	111	ΥΔΡΑΥΛΙΚΟΣ	6740,00	10000,00			M
318	M	A	41	20	3000,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	60000,00	9000,00			A
319	M	A	50	10	4000,00	40	ΖΩΓΡΑΦΟΣ	48000,00	10000,00			A
320	M	A	48	12	15300,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	55000,00	20000,00			A
321	M	A	52	13	1500,00	20	ΓΟΥΝΟΠΟΙΟΣ	48000,00	4000,00			A
322	M	Γ	56	9	3000,00	86	ΟΙΚΙΑΚΑ	4300,00	3000,00			A
323	M	A	61	4	4350,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	4400,00	4000,00			A
324	M	A	58	7	5100,00	95	ΡΑΠΤΗΣ	4300,00	3000,00			A
325	M	A	63	2	5000,00	47	ΙΕΡΕΑΣ	12000,00	5000,00			A
326	M	A	58	7	5900,00	115	ΦΑΝΟΠΟΙΟΣ	13800,00	10000,00			A
327	M	A	59	6	4310,00	7	ΑΡΤΟΠΟΙΟΣ	9800,00	8000,00			A
328	M	A	56	9	4800,00	16	ΓΑΛΑΚΤΟΚΟΜΟΣ	9120,00	5000,00			A
329	M	Γ	56	9	1800,00	31	ΕΚΔΟΤΗΣ	45120,00	5000,00			A
330	M	Γ	59	6	3500,00	46	ΙΑΤΡΟΣ	48300,00	10000,00			A
331	M	A	60	5	18000,00	32	ΕΚΠΑΙΔΕΥΤΙΚΟΣ	18300,00	15000,00			A
332	M	A	60	5	19000,00	124	ΧΡΗΜΑΤΙΣΤΗΣ	46000,00	25000,00			A
333	M	Γ	60	5	14300,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	46350,00	15000,00			A
334	M	A	61	4	12000,00	120	ΦΩΤΟΓΡΑΦΟΣ	61800,00	10000,00			A
335	M	A	60	5	47000,00	70	ΜΗΧΑΝΙΚΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	35100,00	20000,00	Medium	0,420	M

336	M	A	56	9	1500,00	1	ΑΓΡΟΤΗΣ	3500,00	4000,00			M
337	M	A	56	9	1600,00	85	ΟΙΚΟΔΟΜΟΣ	13100,00	4000,00			A
338	M	Γ	22	30	20300,00	72	ΜΟΝΤΕΛΟ	50000,00	10000,00			A
339	M	A	18	20	20800,00	107	ΤΡΑΓΟΥΔΙΣΤΗΣ	48000,00	20000,00			A
340	M	A	20	25	3800,00	53	ΓΡΑΦΕΙΟ ΚΗΔΕΙΩΝ	47000,00	8300,00			A
341	M	A	22	23	5500,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	46500,00	10000,00			M
342	M	Γ	18	22	13100,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	43800,00	15000,00			A
343	M	A	18	32	45300,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	4300,00	4000,00	Good	0,860	A
344	M	Γ	20	30	4600,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	14800,00	10000,00			A
345	M	Γ	28	32	24100,00	71	ΜΟΔΙΣΤΡΑ	12300,00	8000,00			M
346	M	A	35	30	28300,00	47	ΙΕΡΕΑΣ	14800,00	12000,00			A
347	M	A	29	20	5300,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	46800,00	10600,00			A
348	M	A	32	10	2800,00	105	ΤΟΠΟΓΡΑΦΟΣ	51800,00	5000,00			A
349	M	A	32	33	5000,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	47000,00	15000,00			A
350	M	A	27	28	12000,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	47300,00	20000,00			A
351	M	Γ	26	30	9300,00	55	ΚΟΜΜΩΤΡΙΑ	45050,00	15000,00			A
352	M	A	35	30	1500,00	21	ΓΡΑΦΙΣΤΑΣ	45830,00	4500,00			A
353	M	A	32	20	1200,00	28	ΔΙΚΗΓΟΡΟΣ	48000,00	3000,00			A
354	M	A	37	10	1600,00	13	ΒΙΟΜΗΧΑΝΟΣ	50000,00	4800,00			A
355	M	A	42	20	25000,00	105	ΤΟΠΟΓΡΑΦΟΣ	25600,00	10000,00			A
356	M	A	41	10	20300,00	21	ΓΡΑΦΙΣΤΑΣ	38300,00	10000,00			A
357	M	A	50	15	22100,00	28	ΔΙΚΑΣΤΗΣ	46800,00	20000,00			A
358	M	A	51	14	20500,00	41	ΗΘΟΠΟΙΟΣ	48100,00	15000,00			A
359	M	Γ	48	12	46100,00	28	ΔΙΚΗΓΟΡΟΣ	38300,00	10300,00	Bad	0,278	M
360	M	A	48	17	45500,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	40120,00	10000,00	Medium	0,602	A
361	M	A	47	10	50000,00	40	ΖΩΓΡΑΦΟΣ	56000,00	10000,00	Good	0,696	A
362	M	Γ	49	11	48100,00	28	ΔΙΚΑΣΤΗΣ	48350,00	12000,00	Bad	0,348	M
363	M	Γ	50	15	49000,00	105	ΤΟΠΟΓΡΑΦΟΣ	50100,00	12000,00	Bad	0,304	M
364	M	A	42	10	46700,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	48000,00	11000,00	Bad	0,330	M
365	M	A	56	5	21300,00	84	ΟΔΟΝΤΟΤΕΧΝΙΤΗΣ	45300,00	8000,00			A
366	M	A	61	4	28100,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	49130,00	15000,00	Medium	0,540	A
367	M	A	58	7	25380,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	50000,00	12500,00	Good	0,740	A
368	M	Γ	61	4	29350,00	91	ΠΟΛΙΤΙΚΟΣ ΜΗΧΑΝΙΚΟΣ	45010,00	20000,00	Medium	0,460	M
369	M	Γ	56	5	4800,00	87	ΟΠΩΡΟΠΩΛΗΣ	16800,00	10000,00			A
370	M	Γ	60	5	5800,00	39	ΖΑΧΑΡΟΠΛΑΣΤΗΣ	15350,00	12000,00			A
371	M	A	59	6	6000,00	42	ΗΛΕΚΤΡΟΛΟΓΟΣ	19100,00	18000,00			A
372	M	Γ	22	20	3800,00	1	ΑΓΡΟΤΗΣ	16100,00	15000,00			A
373	M	A	20	15	5500,00	42	ΗΛΕΚΤΡΟΛΟΓΟΣ	23100,00	5500,00			A
374	M	A	21	30	4800,00	126	ΨΥΚΤΙΚΟΣ	19880,00	12000,00			A
375	M	A	20	30	5900,00	111	ΥΔΡΑΥΛΙΚΟΣ	18740,00	18000,00			A
376	M	Γ	18	12	45500,00	35	ΕΜΠΟΡΟΣ ΚΑΤ ΥΛΙΚΩΝ	53800,00	20000,00	Good	0,860	A
377	M	A	19	20	46000,00	85	ΕΡΓΟΛΑΒΟΣ ΟΙΚΟΔΟΜΩΝ	47300,00	10000,00	Good	0,870	A
378	M	A	22	12	18500,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	18800,00	8000,00			A
379	M	A	21	15	1680,00	93	ΠΥΡΟΣΒΕΣΤΗΣ	16300,00	4800,00			A
380	M	Γ	19	31	1100,00	112	ΙΔ ΥΠΑΛΛΗΛΟΣ	15230,00	3300,00			A
381	M	Γ	25	30	7300,00	86	ΟΙΚΙΑΚΑ	4500,00	2000,00			A
382	M	A	28	30	10000,00	70	ΜΗΧΑΝΙΚΟΣ ΑΥΤΟΚΙΝΗΤΩΝ	4300,00	4000,00			A
383	M	A	40	18	12100,00	98	ΣΤΡΑΤΙΩΤΙΚΟΣ	4380,00	5000,00			M
384	M	A	38	20	48000,00	62	ΛΟΓΙΣΤΗΣ	18400,00	11000,00	Good	0,740	A
385	M	A	35	25	49000,00	81	ΟΔΗΓΟΣ ΤΑΞΙ	15340,00	5000,00	Medium	0,540	A

386	M	A	23	32	46000,00	34	ΕΜΠΟΡΟΣ	20000,00	10000,00	Good	0,870	A
387	M	A	40	10	50000,00	9	ΑΡΧΙΤΕΚΤΟΝΑΣ	40000,00	7000,00	Medium	0,600	A
388	M	A	35	15	1680,00	85	ΟΙΚΟΔΟΜΟΣ	3800,00	3500,00			A
389	M	A	34	16	1500,00	7	ΑΡΤΟΠΟΙΟΣ	4350,00	2000,00			A
390	M	A	29	11	1300,00	54	ΚΗΠΟΥΡΟΣ	4130,00	3900,00			A
391	M	A	24	15	1100,00	5	ΑΝΘΟΠΩΛΗΣ	4100,00	3000,00			A
392	M	A	25	10	1600,00	1	ΑΓΡΟΤΗΣ	4130,00	4000,00			A
393	M	A	31	20	1550,00	42	ΗΛΕΚΤΡΟΛΟΓΟΣ	4000,00	4000,00			A

Παράρτημα ΙΙ

Κώδικας Προγράμματος σε FuzzyCLIPS

```
;----------------------------------------ORISMOS GLOBAL METABLHTON
(defglobal
?*Sex* = none
?*DiarkeiaAsfalishs* = 0
?*Age* = 5
?*DiarkeiaAsfalishs1* = 0
?*Age1* = 5
?*q* = 0.0
?*b* = 0
?*poso* = 0.0
?*rateI*  = 0.0
?*rateA*  = 0.0
?*rateM*  = 0.0
?*EpilPosoAsfal* = 900
?*BasikoAsfalistroZwhs* = 0.0
?*BasikoAsfalistroPrAtyx* = 0.0
?*EidosAsfalias* = 0
?*PrAtyxhma* = 0.0
?*Job* = 0
?*Eisodhma* = 2500
?*KefPrAt* = 0
?*KefPrAt1* = 0
?*Danger* = 0
?*TestSet* = 0
?*Score* = -1
?*test1* = 0
?*test2* = 0
?*test3* = 0
?*test4* = 0
?*test5* = 0
?*test6* = 0
?*test7* = 0
?*test8* = 0
?*test9* = 0
?*test10* = 0
?*test11* = 0
?*Asfalistra* = 0.0
?*Dosh1* = 0
?*Dosh2* = 0
?*Dosh3* = 0
?*AA* = 0
?*cf1* = 0.0
?*cf2* = 0.0
?*Asfalish* = 5
?*AsfalishEstimation* = 5
)
```

```
;---------------------------------------DEFTEMPLATES
(deftemplate periptwseis
(slot AA (type INTEGER) (default ?NONE))
(slot EidosAsfalias (type INTEGER) (default ?NONE))
(slot Sex (type SYMBOL) (default ?NONE))
(slot Age1 (type INTEGER)(default ?NONE))
(slot DiarkeiaAsfalishs1 (type INTEGER) (default ?NONE))
(slot EpilPosoAsfal (type INTEGER) (default ?NONE))
(slot Job (type INTEGER)(default ?NONE))
(slot test1 (type INTEGER) (default ?NONE))
(slot test2 (type INTEGER) (default ?NONE))
(slot test3 (type INTEGER) (default ?NONE))
(slot test4 (type INTEGER) (default ?NONE))
(slot test5 (type INTEGER) (default ?NONE))
(slot test6 (type INTEGER) (default ?NONE))
(slot test7 (type INTEGER) (default ?NONE))
(slot test8 (type INTEGER) (default ?NONE))
(slot test9 (type INTEGER) (default ?NONE))
(slot test10 (type INTEGER) (default ?NONE))
(slot test11 (type INTEGER) (default ?NONE))
(slot Eisodhma (type INTEGER)(default ?NONE))
(slot KefPrAt1 (type INTEGER)(default ?NONE))
(slot Asfalish (type INTEGER) (default ?NONE))
)

(deftemplate periptwseisFINAL
(slot AA (type INTEGER) (default ?NONE))
(slot EidosAsfalias (type INTEGER) (default ?NONE))
(slot Sex (type SYMBOL) (default ?NONE))
(slot Age (type INTEGER)(default ?NONE))
(slot DiarkeiaAsfalishs (type INTEGER) (default ?NONE))
(slot EpilPosoAsfal (type INTEGER) (default ?NONE))
(slot Job (type INTEGER)(default ?NONE))
(slot test1 (type INTEGER) (default ?NONE))
(slot test2 (type INTEGER) (default ?NONE))
(slot test3 (type INTEGER) (default ?NONE))
(slot test4 (type INTEGER) (default ?NONE))
(slot test5 (type INTEGER) (default ?NONE))
(slot test6 (type INTEGER) (default ?NONE))
(slot test7 (type INTEGER) (default ?NONE))
(slot test8 (type INTEGER) (default ?NONE))
(slot test9 (type INTEGER) (default ?NONE))
(slot test10 (type INTEGER) (default ?NONE))
(slot test11 (type INTEGER) (default ?NONE))
(slot Eisodhma (type INTEGER)(default ?NONE))
(slot KefPrAt (type INTEGER)(default ?NONE))
)

(deftemplate apotelesmata
(slot AA (type INTEGER) (default ?NONE))
(slot cf1 (type FLOAT)(default ?NONE))
(slot cf2 (type FLOAT)(default ?NONE))
(slot Asfalistra (type FLOAT)(default ?NONE))
(slot Asfalish (type INTEGER) (default ?NONE))
```

```
(slot AsfalishEstimation (type INTEGER) (default ?NONE))
)

(deftemplate res1
(slot AA (type INTEGER) (default ?NONE))
(slot Danger (type INTEGER)(default ?NONE))
(slot Score (type INTEGER)(default ?NONE))
(slot BasikoAsfalistroZwhs (type FLOAT)(default ?NONE))
)



;----------------------------------------KANONES ARXEIWN
(defrule open_file1
(declare (salience 280))
=>
(open "arxikopoihsh_gegonotwn_A.txt" arxikopoihsh_gegonotwn_A "r")
(load-facts "arxikopoihsh_gegonotwn_A.txt")
(assert (readvalues))
(open "arxikopoihsh_gegonotwn_A_FINAL.txt" arxikopoihsh_gegonotwn_A_FINAL "w")
(open "results_part1.txt" results_part1 "w")
(open "results_part2.txt" results_part2 "w")
)


;----------------------------------------DHLOSH HLIKIAS
(deffunction elegxos_hlikias (?r ?s)
(if (and (or (eq ?s 1)(eq ?s 2)(eq ?s 4)(eq ?s 5))(< ?r 16)) then
(bind ?*Age* 16)
else (if (and (or (eq ?s 3)(eq ?s 6))(< ?r 20)) then
(bind ?*Age* 20)
else
(bind ?*Age* ?r)
(assert (Age ?r))
))
(assert (Age ?*Age*))
)

;----------------------------------------DIARKEIA ASFALISHS
(deffunction elegxos_asfalishs (?o ?p)
(if (> (+ ?o ?p) 65) then
(bind ?*DiarkeiaAsfalishs* (- 65 ?o))
else
(bind ?*DiarkeiaAsfalishs* ?p)
(bind ?*DiarkeiaAsfalishs1* 0)
)
(assert (DiarkeiaAsfalishs ?*DiarkeiaAsfalishs*))
)

;----------------------------------------EPAGGELMA
(deffunction elegxos_epagelmatos (?i ?j)
(if (or (eq ?i 3)(eq ?i 5)(eq ?i 6)(eq ?i 8)(eq ?i 9)(eq ?i 10)(eq ?i 11)(eq ?i 12)(eq ?i 13)(eq ?i
14)(eq ?i 15)(eq ?i 17)
(eq ?i 18)(eq ?i 19)(eq ?i 20)(eq ?i 21)(eq ?i 22)(eq ?i 24)(eq ?i 26)(eq ?i 27)(eq ?i 28)(eq ?i
```

```
30)(eq ?i 31)(eq ?i 32)
(eq ?i 34)(eq ?i 37)(eq ?i 38)(eq ?i 40)(eq ?i 41)(eq ?i 43)(eq ?i 44)(eq ?i 45)(eq ?i 46)(eq ?i
47)(eq ?i 48)(eq ?i 49)
(eq ?i 50)(eq ?i 51)(eq ?i 52)(eq ?i 53)(eq ?i 54)(eq ?i 55)(eq ?i 56)(eq ?i 60)(eq ?i 62)(eq ?i
64)(eq ?i 69)(eq ?i 71)
(eq ?i 72)(eq ?i 73)(eq ?i 74)(eq ?i 77)(eq ?i 78)(eq ?i 83)(eq ?i 84)(eq ?i 86)(eq ?i 89)(eq ?i
91)(eq ?i 92)(eq ?i 94)
(eq ?i 95)(eq ?i 96)(eq ?i 97)(eq ?i 99)(eq ?i 101)(eq ?i 102)(eq ?i 103)(eq ?i 104)(eq ?i
105)(eq ?i 107)(eq ?i 108)
(eq ?i 112)(eq ?i 116)(eq ?i 117)(eq ?i 119)(eq ?i 120)(eq ?i 122)(eq ?i 124)(eq ?i 127)) then
(bind ?*Danger* 1)
(bind ?*KefPrAt* ?j)
else (if (or (eq ?i 1)(eq ?i 7)(eq ?i 16)(eq ?i 23)(eq ?i 25)(eq ?i 33)(eq ?i 35)(eq ?i 36)(eq ?i
39)(eq ?i 42)(eq ?i 57)
(eq ?i 58)(eq ?i 59)(eq ?i 61)(eq ?i 63)(eq ?i 65)(eq ?i 66)(eq ?i 67)(eq ?i 68)(eq ?i 70)(eq ?i
75)(eq ?i 79)(eq ?i 80)
(eq ?i 81)(eq ?i 85)(eq ?i 87)(eq ?i 90)(eq ?i 93)(eq ?i 100)(eq ?i 106)(eq ?i 109)(eq ?i
110)(eq ?i 111)(eq ?i 113)
(eq ?i 114)(eq ?i 115)(eq ?i 118)(eq ?i 121)(eq ?i 123)(eq ?i 125)(eq ?i 126)) then
(bind ?*Danger* 2)
(bind ?*KefPrAt* ?j)
else (if (or (eq ?i 2)(eq ?i 4)(eq ?i 29)(eq ?i 76)(eq ?i 82)(eq ?i 88)(eq ?i 98)) then
(bind ?*Danger* 3)
(bind ?*KefPrAt* 0)
)))
(bind ?*Job* ?i)
(assert (Job ?*Job*))
(assert (Danger ?*Danger*))
(assert (KefPrAt ?*KefPrAt*))
)

;----------------------------------------IATRIKES EXETASEIS
(deffunction Medical_Tests (?l ?m)
(if (or (<= ?l 22500)
(and (>= ?m 16)(<= ?m 55)(> ?l 22500)(<= ?l 30000))
(and (>= ?m 16)(<= ?m 45)(> ?l 30000)(<= ?l 45000))) then
(bind ?*TestSet* 0)
else (if (or (and (>= ?m 16)(<= ?m 45)(> ?l 45000)(<= ?l 60000))
(and (>= ?m 46)(<= ?m 55)(> ?l 30000)(<= ?l 45000))
(and (>= ?m 56)(<= ?m 65)(> ?l 22500)(<= ?l 30000))) then
(bind ?*TestSet* 1)
else (if (or (and (>= ?m 56)(<= ?m 65)(> ?l 30000)(<= ?l 45000))
(and (>= ?m 16)(<= ?m 45)(> ?l 60000)(<= ?l 90000))) then
(bind ?*TestSet* 2)
else (if (and (>= ?m 46)(<= ?m 65)(> ?l 45000)(<= ?l 90000)) then
(bind ?*TestSet* 3)
else (if (> ?l 90000) then
(bind ?*TestSet* 4)
)))))
)

(deffunction katallhlothta_ygeias (?t1 ?t2 ?t3 ?t4 ?t5 ?t6 ?t7 ?t8 ?t9 ?t10 ?t11)
(if (and (eq ?t4 0)(eq ?t5 0)(eq ?t6 0)(eq ?t7 0)(eq ?t8 0)(eq ?t9 0)(eq ?t10 0)(eq ?t11 0))
then
```

```
(bind ?*Score* (* 20 (+ (* ?t1 0.35)(* ?t2 0.35)(* ?t3 0.3))))
else (if (and (eq ?t6 0)(eq ?t7 0)(eq ?t8 0)(eq ?t9 0)(eq ?t10 0)(eq ?t11 0)) then
(bind ?*Score* (* 20 (+ (* ?t1 0.2)(* ?t2 0.25)(* ?t3 0.19)(* ?t4 0.18)(* ?t5 0.18))))
else (if (and (eq ?t9 0)(eq ?t10 0)(eq ?t11 0)) then
(bind ?*Score* (* 20 (+ (* ?t1 0.13)(* ?t2 0.13)(* ?t3 0.1)(* ?t4 0.12)(* ?t5 0.13)
(* ?t6 0.13)(* ?t7 0.13)(* ?t8 0.13))))
else (if (and (neq ?t1 0)(neq ?t2 0)(neq ?t3 0)(neq ?t4 0)(neq ?t5 0)(neq ?t6 0)(neq ?t7 0)
(neq ?t8 0)(neq ?t9 0)(neq ?t10 0)(neq ?t11 0)) then
(bind ?*Score* (* 20 (+ (* ?t1 0.08)(* ?t2 0.1)(* ?t3 0.08)(* ?t4 0.08)(* ?t5 0.08)
(* ?t6 0.07)(* ?t7 0.1)(* ?t8 0.1)(* ?t9 0.1)(* ?t10 0.15)(* ?t11 0.06))))
else
(bind ?*Score* 0)
))))
(assert (Score ?*Score*))
)

(deffunction Medical_Test_Results (?l ?t1 ?t2 ?t3 ?t4 ?t5 ?t6 ?t7 ?t8 ?t9 ?t10 ?t11)
(if (eq ?l 1) then
(katallhlothta_ygeias ?t1 ?t2 ?t3 0 0 0 0 0 0 0 0)
else (if (eq ?l 2) then
(katallhlothta_ygeias ?t1 ?t2 ?t3 ?t4 ?t5 0 0 0 0 0 0)
else (if (eq ?l 3) then
(katallhlothta_ygeias ?t1 ?t2 ?t3 ?t4 ?t5 ?t6 ?t7 ?t8 0 0 0)
else (if (eq ?l 4) then
(katallhlothta_ygeias ?t1 ?t2 ?t3 ?t4 ?t5 ?t6 ?t7 ?t8 ?t9 ?t10 ?t11)
))))
)

;---------------------------------------PROSTHETH ASFALISH PROSOPIKOY ATYXHMATOS
(deffunction elegxos_KefPrAt (?k ?l ?u ?v ?w)
(bind ?*EpilPosoAsfal* ?w)
(if (or (eq ?k 1)(eq ?k 2)) then
(if (or (eq ?l 4)(eq ?l 5)(eq ?l 6)) then
(if (> ?v ?u) then
(bind ?*KefPrAt* 0)
(bind ?*EpilPosoAsfal* 0)
else (if (> ?v 200000) then
(bind ?*KefPrAt* 0)
(bind ?*EpilPosoAsfal* 0)
else (if (< (* 4 ?w) ?v) then
(bind ?*KefPrAt* 0)
(bind ?*EpilPosoAsfal* 0)
)))
else (if (or (eq ?l 1)(eq ?l 2)(eq ?l 3)) then
(bind ?*KefPrAt* 0)
(bind ?*EpilPosoAsfal* 0)
else
(bind ?*KefPrAt* ?v)
(bind ?*EpilPosoAsfal* ?w)
))
)
(assert (KefPrAt ?*KefPrAt*))
(assert (EpilPosoAsfal ?*EpilPosoAsfal*))
)
```

```
;---------------------------------------MAIN PROGRAM 1
(defrule MainProgram250
(declare (salience 250))
(periptwseis (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age1
?Age1)(DiarkeiaAsfalishs1 ?DiarkeiaAsfalishs1)(EpilPosoAsfal ?EpilPosoAsfal)(Job
?Job)(test1 ?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7
?test7)(test8 ?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma
?Eisodhma)(KefPrAt1 ?KefPrAt1)(Asfalish ?Asfalish))
=>
(bind ?*AA* ?AA) (assert (AA ?*AA*))
(bind ?*Sex* ?Sex) (assert (Sex ?*Sex*))
(bind ?*EidosAsfalias* ?EidosAsfalias) (assert (EidosAsfalias ?*EidosAsfalias*))
(bind ?*Asfalish* ?Asfalish) (assert (Asfalish ?*Asfalish*))
(elegxos_hlikias ?Age1 ?EidosAsfalias)
(elegxos_asfalishs ?*Age* ?DiarkeiaAsfalishs1)
(elegxos_epagelmatos ?Job ?KefPrAt1)
(Medical_Tests ?EpilPosoAsfal ?*Age*)
(Medical_Test_Results ?*TestSet* ?test1 ?test2 ?test3 ?test4 ?test5 ?test6 ?test7 ?test8
?test9 ?test10 ?test11)
(elegxos_KefPrAt ?*Danger* ?EidosAsfalias ?Eisodhma ?KefPrAt1 ?EpilPosoAsfal)
(assert (periptwseisFINAL (AA ?AA)(EidosAsfalias ?*EidosAsfalias*)(Sex ?*Sex*)(Age
?*Age*)(DiarkeiaAsfalishs ?*DiarkeiaAsfalishs*)(EpilPosoAsfal ?*EpilPosoAsfal*)(Job
?*Job*)(test1 ?*test1*)(test2 ?*test2*)(test3 ?*test3*)(test4 ?*test4*)(test5 ?*test5*)(test6
?*test6*)(test7 ?*test7*)(test8 ?*test8*)(test9 ?*test9*)(test10 ?*test10*)(test11
?*test11*)(Eisodhma ?*Eisodhma*)(KefPrAt ?*KefPrAt*)))
)


(defrule Print_arxikopoihsh_gegonotwn_A_FINAL
(declare (salience 250))
(periptwseis (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age1
?Age1)(DiarkeiaAsfalishs1 ?DiarkeiaAsfalishs1)(EpilPosoAsfal ?EpilPosoAsfal)(Job
?Job)(test1 ?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7
?test7)(test8 ?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma
?Eisodhma)(KefPrAt1 ?KefPrAt1)(Asfalish ?Asfalish))
=>
(printout arxikopoihsh_gegonotwn_A_FINAL "(periptwseisFINAL (AA " ?*AA*
")(EidosAsfalias " ?*EidosAsfalias* ")(Sex " ?*Sex* ")(Age "

?*Age* ")(DiarkeiaAsfalishs " ?*DiarkeiaAsfalishs* ")(EpilPosoAsfal " ?*EpilPosoAsfal* ")(Job
" ?*Job* ")(test1 " ?*test1* ")(test2 " ?*test2* ")(test3 " ?*test3* ")(test4 " ?*test4* ")(test5 "
?*test5* ")(test6 " ?*test6* ")(test7 " ?*test7* ")(test8 " ?*test8* ")(test9 " ?*test9* ")(test10 "
?*test10* ")(test11 " ?*test11* ")(Eisodhma " ?*Eisodhma* ")(KefPrAt " ?*KefPrAt* "))" crlf)
(printout results_part1 "(periptwseisFINAL (AA " ?*AA* ")(EidosAsfalias " ?*EidosAsfalias*
")(Sex " ?*Sex* ")(Age " ?*Age* ")(DiarkeiaAsfalishs " ?*DiarkeiaAsfalishs* ")(EpilPosoAsfal "
?*EpilPosoAsfal* ")(Job " ?*Job* ")(test1 " ?*test1* ")(test2 " ?*test2* ")(test3 " ?*test3*
")(test4 " ?*test4* ")(test5 " ?*test5* ")(test6 " ?*test6* ")(test7 " ?*test7* ")(test8 " ?*test8*
")(test9 " ?*test9* ")(test10 " ?*test10* ")(test11 " ?*test11* ")(Eisodhma " ?*Eisodhma*
")(KefPrAt " ?*KefPrAt* "))" crlf)
(printout arxikopoihsh_gegonotwn_A_FINAL "(res1 (AA " ?*AA* ")(Danger " ?*Danger*
")(Score " ?*Score* ")(BasikoAsfalistroZwhs 0.0))"

crlf)
```

```
(bind ?*Score* 0)
)


;----------------------------------------TIMOLOGHSH 1. APLHS
(deffunction Timologhsh_AplhsOrDiarkeias (?i ?j ?k ?x ?z)
(if (eq ?k m) then
(if (and (>= ?j 5)(< ?j 10)) then
(bind ?*q* 28)
else (if (and (>= ?j 10)(< ?j 15)) then
(bind ?*q* 29.70)
else (if (and (>= ?j 15)(< ?j 20)) then
(bind ?*q* 29.83)
else (if (and (>= ?j 20)(< ?j 25)) then
(bind ?*q* 30.12)
else (if (and (>= ?j 25)(< ?j 30)) then
(bind ?*q* 30.97)
else (if (> ?j 30) then
(bind ?*q* 32.35)
))))))
else (if (eq ?k f) then
(if (and (>= ?j 5)(< ?j 10)) then
(bind ?*q* 21.62)
else (if (and (>= ?j 10)(< ?j 15)) then
(bind ?*q* 22.30)
else (if (and (>= ?j 15)(< ?j 20)) then
(bind ?*q* 22.72)
else (if (and (>= ?j 20)(< ?j 25)) then
(bind ?*q* 23.05)
else (if (and (>= ?j 25)(< ?j 30)) then
(bind ?*q* 23.52)
else (if (and (>= ?j 30)(< ?j 35)) then
(bind ?*q* 24.36)
else (if (> ?j 35) then
(bind ?*q* 25.54)
)))))))
))
(assert (q ?*q*))
(while (and (> ?i 15) (< ?i (+ ?x 1))) do
(if (and (> ?i 16) (< ?i 21)) then
(bind ?*rateA* 0.013)
else (if (and (> ?i 20) (< ?i 31)) then
(bind ?*rateA* 0.015)
else (if (and (> ?i 30) (< ?i 41)) then
(bind ?*rateA* 0.019)else (if (and (> ?i 40) (< ?i 51)) then
(bind ?*rateA* 0.05)
else (if (and (> ?i 50) (< ?i 61)) then
(bind ?*rateA* 0.09)
else (if (and (> ?i 60) (< ?i 66)) then
(bind ?*rateA* 0.12)
))))))
(bind ?*poso* (+ ?*q* (* ?*q* ?*rateA*)))
(bind ?*q* ?*poso*)
(bind ?i (+ ?i 1))
```

```
)
(bind ?*BasikoAsfalistroZwhs* (* ?*q* ?z 0.0001))
(assert (BasikoAsfalistroZwhs ?*BasikoAsfalistroZwhs*))
)

;----------------------------------------TIMOLOGHSH 2. MIKTHS
(deffunction Timologhsh_MikthsOrDiarkeias (?u ?v ?w ?x ?z)
(if (eq ?w m) then
(if (and (>= ?v 10)(< ?v 15)) then
(bind ?*q* 96.61)
else (if (and (>= ?v 15)(< ?v 20)) then
(bind ?*q* 61.16)
else (if (and (>= ?v 20)(< ?v 25)) then
(bind ?*q* 43.68)
else (if (and (>= ?v 25)(< ?v 30)) then
(bind ?*q* 33.39)
else (if (> ?v 30) then
(bind ?*q* 26.24)
)))))
else (if (eq ?w f) then
(if (and (>= ?v 10)(< ?v 15)) then
(bind ?*q* 96.26)
else (if (and (>= ?v 15)(< ?v 20)) then
(bind ?*q* 60.76)
else (if (and (>= ?v 20)(< ?v 25)) then
(bind ?*q* 43.24)
else (if (and (>= ?v 25)(< ?v 30)) then
(bind ?*q* 32.92)
else (if (and (>= ?v 30)(< ?v 35)) then
(bind ?*q* 25.74)
)))))
))
(assert (q ?*q*))
(while (and (> ?u 15) (< ?u (+ ?x 1))) do
(if (and (> ?u 16) (< ?u 21)) then
(bind ?*rateM* 0.0002)
else (if (and (> ?u 20) (< ?u 31)) then
(bind ?*rateM* 0.00021)
else (if (and (> ?u 30) (< ?u 41)) then
(bind ?*rateM* 0.00022)
else (if (and (> ?u 40) (< ?u 51)) then
(bind ?*rateM* 0.00023)
else (if (and (> ?u 50) (< ?u 61)) then
(bind ?*rateM* 0.00024)
else (if (and (> ?u 60) (< ?u 66)) then
(bind ?*rateM* 0.00025)
))))))
(bind ?*poso* (+ ?*q* (* ?*q* ?*rateM*)))
(bind ?*q* ?*poso*)
(bind ?u (+ ?u 1))
)
(assert (rateM ?*rateM*))
(bind ?*BasikoAsfalistroZwhs* (* ?*q* ?z 0.0001))
(assert (BasikoAsfalistroZwhs ?*BasikoAsfalistroZwhs*))
```

```
)

;----------------------------------------TIMOLOGHSH 3. ISOBIAS
(deffunction Timologhsh_Isobias (?i ?k ?m)
(if (eq ?k m) then
(bind ?*q* 30)
else (if (eq ?k f) then
(bind ?*q* 22)
))
(assert (q ?*q*))
(bind ?*rateI* 0.025)
(while (and (> ?i 19) (< ?i (+ ?*Age* 1))) do
(bind ?*poso* (+ ?*q* (* ?*q* ?*rateI*)))
(bind ?*q* ?*poso*)
(bind ?i (+ ?i 1))
)
(bind ?*BasikoAsfalistroZwhs* (* ?*q* ?m 0.0001))
(assert (BasikoAsfalistroZwhs ?*BasikoAsfalistroZwhs*))
)

;-----------------------------------------MAIN PROGRAM 2
(defrule MainProgram240
"Ypologismos ethsioy asfalistroy"
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
=>
(if (or (eq ?*Danger* 1)(eq ?*Danger* 2)) then
(if (and (eq ?*EidosAsfalias* 4)(neq ?*DiarkeiaAsfalishs* 0)) then
(Timologhsh_AplhsOrDiarkeias 16 ?*DiarkeiaAsfalishs* ?*Sex* ?*Age* ?*EpilPosoAsfal*)
else (if (and (eq ?*EidosAsfalias* 5) (neq ?*DiarkeiaAsfalishs* 0)) then
(Timologhsh_MikthsOrDiarkeias 16 ?*DiarkeiaAsfalishs* ?*Sex* ?*Age* ?*EpilPosoAsfal*)
else (if (eq ?*EidosAsfalias* 6) then
(Timologhsh_Isobias  20 ?*Sex* ?*EpilPosoAsfal*)
)))
)
)

(defrule Print_res_part1
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
=>
(if (eq ?*Danger* 3) then
(bind ?*BasikoAsfalistroZwhs* 0.0)
(assert (BasikoAsfalistroZwhs ?*BasikoAsfalistroZwhs*))
)
(printout results_part1 "(res1 (AA " ?*AA* ")(Danger " ?*Danger* ")(Score " ?*Score*
```

```
")(BasikoAsfalistroZwhs " ?*BasikoAsfalistroZwhs*

"))" crlf)
(printout results_part1 "(apotelesmata (AA " ?*AA* ")(cf1 " ?*cf1* ")(cf2 " ?*cf2* ")(Asfalistra
?Asfalistra)(Asfalish

?Asfalish)(AsfalishEstimation ?AsfalishEstimation))" crlf)
)


;----------------------------------------DEFTEMPLATES FUZZIFYING
(deftemplate hlikia
0 70
((very_young (0 1) (16 1) (22 0))
(young (20 0) (30 1) (45 0))
(old (40 0) (50 1) (60 0))
(very_old (55 0) (70 1)))
)

(deftemplate ygeia
-10 115
((bad (-10 1) (35 1) (45 0))
(medium (40 0) (50 1) (65 0))
(good (60 0) (70 1) (115 1)))
)

(deftemplate poso_asfalishs
-1000 110000
((very_low (-1000 1) (1500 1) (1700 0))
(low (1500 0) (4000 1) (6000 0))
(medium (5000 0) (13000 1) (22000 0))
(high (20000 0) (30000 1) (50000 0))
(very_high (40000 0) (70000 1) (110000 1)))
)

(deftemplate eisodhma
500 110000
((low (500 1) (3000 1) (5000 0))
(low_med (4000 0) (12000 1) (16000 0))
(med_high (14000 0) (30000 1) (50000 0))
(high (40000 0) (80000 1) (110000 1)))
)

(deftemplate katallhlothta_asfalishs1
0.0 1.15
((akatallhlos1 (0.0 1) (0.2 1) (0.4 0))
(metrios1 (0.3 0) (0.4 1) (0.65 0))
(katallhlos1 (0.55 0) (0.75 1) (1.15 1)))
)

(deftemplate katallhlothta_asfalishs2
0.0 1.15
((akatallhlos2 (0.0 1) (0.3 1) (0.45 0))
(metrios2 (0.4 0) (0.5 1) (0.6 0))
```

```
(katallhlos2 (0.55 0) (0.65 1) (1.1 1)))
)


;----------------------------------------DEFRULES FUZZIFYING
(defrule fuzzify_hlikia
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
=>
(bind ?a1 (- ?Age 5))
(bind ?a2 (+ ?Age 5))
(assert (hlikia (?a1 0) (?Age 1) (?a2 0)))
)

(defrule fuzzify_ygeia
(declare (salience 250))
(and (periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(Score ?Score))
=>
(bind ?s1 (- ?Score 10))
(bind ?s2 (+ ?Score 10))
(assert (ygeia (?s1 0) (?Score 1) (?s2 0)))
)


(defrule fuzzify_poso_asfalishs
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
=>
(bind ?p1 (- ?EpilPosoAsfal 880))
(bind ?p2 (+ ?EpilPosoAsfal 880))
(assert (poso_asfalishs (?p1 0) (?EpilPosoAsfal 1) (?p2 0)))
)

(defrule fuzzify_eisodhma
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal

?EpilPosoAsfal)(Job ?Job)(test1 ?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5
?test5)(test6 ?test6)(test7 ?test7)(test8
```

```
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
=>
(bind ?a1 (- ?Eisodhma 1500))
(bind ?a2 (+ ?Eisodhma 1500))
(assert (eisodhma (?a1 0) (?Eisodhma 1) (?a2 0)))
)


;----------------------------------------KANONES FUZZINESS 1
(defrule rule1
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule2
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule3
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs very_low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule4
(declare (salience 250))
```

```
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule5
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule6
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule7
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))
```

```
(defrule rule8
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs low)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule9
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs medium)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule10
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs medium)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule11
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs medium)
(eisodhma low))
=>
(assert (g 1))
```

```
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule12
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs medium)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule13
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule14
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule15
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs high)
(eisodhma low))
```

```
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule16
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule17
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule18
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule19
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
```

```
(poso_asfalishs very_high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule20
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_high)
(eisodhma low))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule21
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule22
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule23
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
```

```
?KefPrAt))
(and (hlikia old)
(poso_asfalishs very_low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule24
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule25
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule26
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule27
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
```

```
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule28
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs low)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule29
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs medium)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule30
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs medium)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule31
(declare (salience 250))
```

```
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs medium)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule32
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs medium)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule33
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule34
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))
```

```
(defrule rule35
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule36
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule37
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule38
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_high)
(eisodhma low_med))
=>
(assert (g 1))
```

```
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule39
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs very_high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule40
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_high)
(eisodhma low_med))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule41
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule42
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_low)
(eisodhma med_high))
```

```
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule43
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(hlikia old)
(and (poso_asfalishs very_low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule44
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule45
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule46
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
```

```
(poso_asfalishs low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule47
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule48
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs low)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule49
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs medium)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule50
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
```

```
?KefPrAt))
(and (hlikia young)
(poso_asfalishs medium)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule51
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs medium)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule52
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs medium)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule53
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule54
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
```

```
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule55
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule56
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule57
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule58
(declare (salience 250))
```

```
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule59
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs very_high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule60
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_high)
(eisodhma med_high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 akatallhlos1)))

(defrule rule61
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))
```

```
(defrule rule62
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule63
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs very_low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule64
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule65
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs low)
(eisodhma high))
=>
(assert (g 1))
```

```
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule66
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule67
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal

?EpilPosoAsfal)(Job ?Job)(test1 ?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5
?test5)(test6 ?test6)(test7 ?test7)(test8

?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule68
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs low)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule69
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
```

```
(and (hlikia very_young)
(poso_asfalishs medium)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule70
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs medium)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule71
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs medium)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule72
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs medium)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule73
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
```

```
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule74
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule75
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule76
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))

(defrule rule77
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
```

```
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(poso_asfalishs very_high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule78
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(poso_asfalishs very_high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule79
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(poso_asfalishs very_high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 katallhlos1)))

(defrule rule80
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(poso_asfalishs very_high)
(eisodhma high))
=>
(assert (g 1))
(assert (katallhlothta_asfalishs1 metrios1)))
```

```
;----------------------------------------KANONES FUZZINESS 2
(defrule rule101
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule102
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule103
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule104
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia bad))
=>
```

```
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule105
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule106
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule107
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule108
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 katallhlos1)
```

```
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule109
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule110
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule111
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule112
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
```

```
(and (hlikia very_old)
(katallhlothta_asfalishs1 katallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule113
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 katallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule114
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 katallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule115
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 katallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule116
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
```

```
(katallhlothta_asfalishs1 katallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule117
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 metrios1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule118
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 metrios1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule119
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 metrios1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule120
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
```

```
(and (hlikia young)
(katallhlothta_asfalishs1 metrios1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule121
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 metrios1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule122
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 metrios1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule123
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 metrios1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule124
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
```

```
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 metrios1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule125
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 metrios1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule126
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 metrios1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule127
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 metrios1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule128
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
```

```
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 metrios1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule129
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 metrios1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule130
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 metrios1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 katallhlos2)))

(defrule rule131
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 metrios1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule132
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
```

```
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 metrios1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule133
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule134
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule135
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_young)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule136
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
```

```
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule137
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule138
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule139
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule140
```

```
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule141
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule142
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia bad))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule143
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia medium))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))
```

```
(defrule rule144
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 akatallhlos1)
(ygeia good))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))

(defrule rule145
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(hlikia very_young)
(katallhlothta_asfalishs1 akatallhlos1)
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule146
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia young)
(katallhlothta_asfalishs1 akatallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))

(defrule rule147
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia old)
(katallhlothta_asfalishs1 akatallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 metrios2)))
```

```
(defrule rule148
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(and (hlikia very_old)
(katallhlothta_asfalishs1 akatallhlos1))
=>
(assert (f 1))
(assert (katallhlothta_asfalishs2 akatallhlos2)))


;----------------------------------------DEFUZZIFICATION
(defrule defuzzification
(declare (salience 250))
(or (Danger 1)(Danger 2))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
?flag <- (f 1)
?f <- (katallhlothta_asfalishs2 ?j)
  =>
(bind ?*cf2* (moment-defuzzify ?f))
(assert (cf2 ?*cf2*))
(retract ?flag)
(assert (f 0))
)

(deffunction Asfalish_Estimation (?x )
(if (>= ?x 0.55) then
(bind ?*AsfalishEstimation* 1)
else (if (< ?x 0.55) then
(bind ?*AsfalishEstimation* 0)
))
)


(defrule defuzzification1
(declare (salience 250))
(and (periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(or (Danger 1)(Danger 2)))
?flag <- (g 1)
?g <- (katallhlothta_asfalishs1 ?i)
=>
(bind ?*cf1* (moment-defuzzify ?g))
(assert (cf1 ?*cf1*))
```

```
(retract ?flag)
)


(defrule print_res_part2
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
=>
(Asfalish_Estimation ?*cf2*)
(printout results_part2 "(apotelesmata (AA " ?*AA* ")(cf2 " ?*cf2* ")(Asfalistra " ?*Asfalistra*
")(Asfalish " ?*Asfalish*")(AsfalishEstimation " ?*AsfalishEstimation* "))" crlf)
)


;----------------------------------------TIMOLOGHSH PROSOPIKOY ATYXHMATOS
(deffunction Timologhsh_PrAtyxhmatos (?i ?k ?a)
(if (and (>= ?a 0.66) (<= ?a 1.1)) then
(if (eq ?k 1) then
(if (and (>= ?i 16)(<= ?i 35)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001))
else (if (and (>= ?i 36)(<= ?i 50)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001 1.05))
else (if (and (>= ?i 51)(<= ?i 60)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001 1.06))
else (if (and (>= ?i 61)(<= ?i 65)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001 1.07))
))))
else (if (eq ?k 2) then
(if (and (>= ?i 16)(<= ?i 35)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001))
else (if (and (>= ?i 36)(<= ?i 50)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001 1.05))
else (if (and (>= ?i 51)(<= ?i 60)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001 1.06))
else (if (and (>= ?i 61)(<= ?i 65)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001 1.07))
))))
))
else (if (and (>= ?a 0.45) (< ?a 0.66)) then
(if (eq ?k 1) then
(if (and (>= ?i 16)(<= ?i 35)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001))
else (if (and (>= ?i 36)(<= ?i 50)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001 1.05))
else (if (and (>= ?i 51)(<= ?i 60)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001 1.06))
else (if (and (>= ?i 61)(<= ?i 65)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 14 0.001 1.07))
))))
else (if (eq ?k 2) then
```

```
(if (and (>= ?i 16)(<= ?i 35)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001))
else (if (and (>= ?i 36)(<= ?i 50)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001 1.05))
else (if (and (>= ?i 51)(<= ?i 60)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001 1.06))
else (if (and (>= ?i 61)(<= ?i 65)) then
(bind ?*BasikoAsfalistroPrAtyx* (* ?*KefPrAt* 20 0.001 1.07))
))))
))
else (if (and (>= ?a 0.0) (< ?a 0.45)) then
(bind ?*BasikoAsfalistroZwhs* 0)
(bind ?*BasikoAsfalistroPrAtyx* 0)
)))
(assert (BasikoAsfalistroPrAtyx ?*BasikoAsfalistroPrAtyx*))
)

(defrule runfunc4
"Ypologismos ethsioy asfalistroy"
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias ?EidosAsfalias)(Sex ?Sex)(Age
?Age)(DiarkeiaAsfalishs ?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1
?test1)(test2 ?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8
?test8)(test9 ?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt
?KefPrAt))
(or (Danger 1)(Danger 2))
=>
(Timologhsh_PrAtyxhmatos ?*Age* ?*Danger* (/ (+ ?*cf1* ?*cf2*) 2))
)

;----------------------------------------RESULTS
(deffunction CalAsfalistra (?k ?l ?m )
(if (eq ?k 0) then
(bind ?*Asfalistra* (+ ?l ?m))
else (if (and (> ?k 0)(<= ?k 40)) then
(bind ?*Asfalistra* 0)
else (if (and (> ?k 40)(<= ?k 50)) then
(bind ?*Asfalistra* (* 1.1 (+ ?l ?m)))
else (if (and (> ?k 50)(<= ?k 65)) then
(bind ?*Asfalistra* (* 1.06 (+ ?l ?m)))
else (if (and (> ?k 65)(<= ?k 80)) then
(bind ?*Asfalistra* (* 1.02 (+ ?l ?m)))
else (if (> ?k 80) then
(bind ?*Asfalistra* (+ ?l ?m))
))))))
(assert (Asfalistra ?*Asfalistra*))
)

(defrule results4
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias 4)(Sex ?Sex)(Age ?Age)(DiarkeiaAsfalishs
?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1 ?test1)(test2
?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8 ?test8)(test9
?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt ?KefPrAt))
```

```
=>
(CalAsfalistra ?*Score* ?*BasikoAsfalistroZwhs* ?*BasikoAsfalistroPrAtyx*)
)

(defrule results5
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias 5)(Sex ?Sex)(Age ?Age)(DiarkeiaAsfalishs
?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1 ?test1)(test2
?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8 ?test8)(test9
?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt ?KefPrAt))
=>
(CalAsfalistra ?*Score* ?*BasikoAsfalistroZwhs* ?*BasikoAsfalistroPrAtyx*)
)

(defrule results6
(declare (salience 250))
(periptwseisFINAL (AA ?AA)(EidosAsfalias 6)(Sex ?Sex)(Age ?Age)(DiarkeiaAsfalishs
?DiarkeiaAsfalishs)(EpilPosoAsfal ?EpilPosoAsfal)(Job ?Job)(test1 ?test1)(test2
?test2)(test3 ?test3)(test4 ?test4)(test5 ?test5)(test6 ?test6)(test7 ?test7)(test8 ?test8)(test9
?test9)(test10 ?test10)(test11 ?test11)(Eisodhma ?Eisodhma)(KefPrAt ?KefPrAt))
=>
(CalAsfalistra ?*Score* ?*BasikoAsfalistroZwhs* ?*BasikoAsfalistroPrAtyx*)
)

(defrule final_printing
(declare (salience 100))
=>
(close)
)
```