ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΥΣΤΗΜΑΤΑ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΕΞΥΠΗΡΕΤΗΣΗΣ ΜΕ ΖΗΤΗΣΗ ΠΟΥ ΕΠΗΡΕΑΖΕΤΑΙ ΑΠΟ ΤΗΝ ΙΚΑΝΟΠΟΙΗΣΗ ΠΕΛΑΤΩΝ

Διατριβή που υπεβλήθη για την μερική ικανοποίηση των απαιτήσεων για την απόκτηση Διδακτορικού Διπλώματος

υπό

Δημητρίου Κωνσταντά

Χανιά, 2025

ΕΠΤΑΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

Συστήματα παραγωγής και εξυπηρέτησης με ζήτηση που επηρεάζεται από την ικανοποίηση πελατών

Production and service systems with demand affected by customer satisfaction

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

Δημήτριος Κωνσταντάς

ΤΡΙΜΕΛΗΣ ΣΥΜΒΟΥΛΕΥΤΙΚΗ ΕΠΙΤΡΟΠΗ:

1. **Ευστράτιος Ιωαννίδης(Επιβλέπων)**
2. **Γεώργιος Αραμπατζής**
3. **Βασίλειος Κουϊκόγλου**

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή την: 03 / 01 / 2025.

Ευστράτιος Ιωαννίδης, Αναπληρωτής Καθηγητής, Πολυτεχνείο Κρήτης

EFSTRATIOS IOANNIDIS
03/01/2025 14:54

Γεώργιος Αραμπατζής, Αναπληρωτής Καθηγητής, Πολυτεχνείο Κρήτης

Georgios Arampatzis
Digitally signed by Georgios Arampatzis
Date: 2025.01.03 15:07:46 +02'00'

Βασίλειος Κουϊκόγλου, Καθηγητής, Πολυτεχνείο Κρήτης

Vasilis Kouikoglou
2025.01.03 13:57:32 +02'00'

Ελευθέριος Δοϊτσίδης, Αναπληρωτής Καθηγητής, Πολυτεχνείο Κρήτης

ELEFTHERIOS DOITSIDIS
03.01.2025 15:39

Δημήτριος Κουλουριώτης, Καθηγητής, Εθνικό Μετσόβιο Πολυτεχνείο

Αλέξανδρος Ξανθόπουλος, Επίκουρος Καθηγητής, Δημοκρίτειο Πανεπιστήμιο Θράκης

Alexandros Xanthopoulos
Alexandros Xanthopoulos
03.01.2025 14:14

Γεώργιος Σαχαρίδης, Αναπληρωτής Καθηγητής, Πανεπιστήμιο Θεσσαλίας

ΠΕΡΙΕΧΟΜΕΝΑ

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τα μέλη της τριμελούς συμβουλευτικής επιτροπής για την καθοδήγηση, τις πολύτιμες υποδείξεις και την συμπαράσταση τους καθόλη την διάρκεια εκπόνησης της διατριβής. Ακόμη οφείλω ένα μεγάλο ευχαριστώ στους γονείς μου για την αμέριστη υποστήριξη τους καθ’ όλη την διάρκεια των σπουδών μου. Τέλος θα ήθελα να κάνω ειδική μνεία στον αξέχαστο Καθηγητή και δάσκαλο Ευάγγελο Γρηγορούδη, η συμβολή του οποίου ήταν καθοριστική για την ολοκλήρωση αυτής της διδακτορικής διατριβής.

Σύντομο βιογραφικό σημείωμα

Ο Δημήτριος Κωνσταντάς γεννήθηκε στην Αθήνα το 1981. Αποφοίτησε από το τμήμα Μηχανικών Παραγωγής και Διοίκησης του Πολυτεχνείου Κρήτης το 2005. Απέκτησε το Μεταπτυχιακό Δίπλωμα Ειδίκευσης του ιδίου τμήματος το 2010. Μέχρι σήμερα είναι υποψήφιος διδάκτορας του τμήματος Μηχανικών Παραγωγής και Διοίκησης. Ερευνητικά ασχολείται με προβλήματα σχεδίασης και ελέγχου συστημάτων παραγωγής με κλασσικές μεθόδους.

Περίληψη

Σε αυτή τη διατριβή παρουσιάζονται μαθηματικά μοντέλα που αναπτύχθηκαν για την κατανόηση του τρόπου με τον οποίο η ποιότητα του προϊόντος μπορεί να επηρεάσει την κερδοφορία των συστημάτων παραγωγής, όταν οι καταναλωτές βασίζουν τα μελλοντικά μοτίβα ζήτησής τους στην ποιότητα του προϊόντος που έχουν πρόσφατα αγοράσει. Εξετάζουμε συστήματα παραγωγής, που λαμβάνουν παραγγελίες από τακτικούς και περιστασιακούς πελάτες, με τους πρώτους να έχουν υψηλότερο μέσο ρυθμό ζήτησης. Κάθε εξερχόμενο προϊόν υπόκειται σε έλεγχο ποιότητας για να αποφασιστεί εάν θα απορριφθεί ως μη συμμορφούμενο ή θα αποσταλεί σε πελάτη. Στην τελευταία περίπτωση, ο πελάτης που αγοράζει το τελικό προϊόν θα γίνει στη συνέχεια τακτικός ή περιστασιακός πελάτης με συμπληρωματικές πιθανότητες που εξαρτώνται από το επίπεδο ποιότητας του συγκεκριμένου προϊόντος. Μελετήσαμε δίκτυα παραγωγής που παράγουν κατά παραγγελίες (make-to-order), συστήματα ενός σταδίου στα οποία η ικανοποίηση πελατών καθορίζεται από την προηγούμενη κατάσταση των πελατών και συστήματα παραγωγής προς αποθεματοποίηση (make-to-stock) ενός σταδίου. Τα αποτελέσματα των αριθμητικών πειραμάτων, που πραγματοποιήθηκαν με δυναμικό προγραμματισμό, δείχνουν ότι η βέλτιστη πολιτική εξαρτάται από την κατάσταση του συστήματος, είναι πολύπλοκη και έχει αυξημένες υπολογιστικές απαιτήσεις. Στην περίπτωση των δικτύων παραγωγής κατά παραγγελίες, χωρίς μνήμη της προηγούμενης κατάστασης των πελατών, προτείνεται μια απλή ευρετική πολιτική τύπου κατωφλίου. Για την ανάλυση και την βελτιστοποίηση της ευρετικής πολιτικής χρησιμοποιούνται μοντέλα κλειστών δικτύων αναμονής με ελάχιστες υπολογιστικές απαιτήσεις. Τα αριθμητικοί αποτελέσματα υποδεικνύουν ότι η προτεινόμενη ευρετική πολιτική αποδίδει σχεδόν το ίδιο καλά με τη βέλτιστη πολιτική.

1 Εισαγωγή

1.1 Αντικείμενο της διατριβής

Η ικανοποίηση των πελατών θεωρείται καθοριστικός παράγοντας της κερδοφορίας και της βιωσιμότητας των βιομηχανικών επιχειρήσεων, λόγω της άμεσης επίδρασής της στο μερίδιο αγοράς και, συνεπώς, στις πωλήσεις και στα έσοδα. Η ικανοποίηση των πελατών είναι μια σύνθετη, δυναμική διαδικασία, που εξαρτάται από πολλούς παράγοντες, όπως η ποιότητα του προϊόντος, η τιμή του προϊόντος, ο χρόνος απόκρισης στις παραγγελίες των πελατών, κ.λπ. Ενώ είναι προφανές ότι υπάρχει άμεση σύνδεση της ικανοποίησης πελατών με την ορθή λειτουργία και τον έλεγχο του συστήματος παραγωγής μιας βιομηχανικής επιχείρησης η αλληλεπίδραση μεταξύ τους έχει μελετηθεί ελάχιστα. Σε αυτή την εργασία μελετάμε την επίδραση του συνδυασμένου ελέγχου παραγωγής και ποιότητας στην ικανοποίηση των πελατών και συνεπώς στην διαμόρφωση της ζήτησης για το παραγόμενο προϊόν. Ο έλεγχος παραγωγής περιλαμβάνει τον χρονικό προγραμματισμό της λειτουργίας κάθε μηχανής του συστήματος (πότε ξεκινά η παραγωγή, με τι ρυθμό και πότε διακόπτεται). Ο έλεγχος ποιότητας ασχολείται με το πρόβλημα του καθορισμού του αποδεκτού επιπέδου ποιότητας των παραγόμενων προϊόντων. Δηλαδή προσπαθεί να απαντήσει στο ερώτημα πότε ένα προϊόν είναι κατάλληλο προς πώληση και πότε πρέπει να επορριφθεί ή να επισκευασθεί/επανακατεργασθεί. Οι απαντήσεις στα προηγούμενα ερωτήματα επηρεάζουν την ικανοποίηση πελατών και κατά συνέπεια το επίπεδο πωλήσεων, το οποίο με την σειρά του έχει σημαντική επίδραση στις αποφάσεις ελέγχου παραγωγής και ποιότητας.

Σε αυτή την εργασία αναπτύχθηκαν μαθηματικά μοντέλα που μας επιτρέπουν να μελετήσουμε την αλληλεπίδραση των αποφάσεων ελέγχου παραγωγής και ποιότητας, με την ικανοποίηση πελατών και τον μέσο ρυθμό πωλήσεων. Πιο συγκεκριμένα χρησιμοποιήσαμε εργαλεία από την θεωρία στοχαστικού δυναμικού προγραμματισμού για να διερευνήσουμε αριθμητικά την δομή της βέλτιστης πολιτικής. Μελετήσαμε τόσο συστήματα παραγωγής κατά παραγγελίες (make-to-order) όσο και συστήματα παραγωγής προς αποθεματοποίηση (make-to-stock) και σε κάποιες περιπτώσεις προτείναμε απλές αλλά αποτελεσματικές ευρετικές πολιτικές, που μπορούν εύκολα να αναλυθούν με την χρήση εργαλείων από την θεωρία συστημάτων αναμονής.

1.2 Βιβλιογραφική ανασκόπηση

Η ικανοποίηση του πελάτη θεωρείται ότι παρέχει στις εταιρείες ανταγωνιστικό πλεονέκτημα για τη διατήρηση και αύξηση του μεριδίου τους στην αγορά, των πωλήσεών τους και των εσόδων τους. Η φύση της ικανοποίησης του πελάτη, οι βασικοί παράγοντές της και οι συνέπειές της έχουν εξεταστεί από τους Anderson et al. [2] και Oliver [32]. Παρόλο που είναι δύσκολο να αρνηθούμε τα οφέλη της, η ιδέα της ενσωμάτωσης της ικανοποίησης του πελάτη στη στρατηγική της εταιρείας αντιμετωπίστηκε αρχικά με πολλή επιφυλακτικότητα από τα διευθυντικά στελέχη. Αυτό οφειλόταν στην έλλειψη συγκεκριμένων και μετρήσιμων αποδεικτικών στοιχείων για την επίδρασή της στην απόδοση των επιχειρήσεων. Σύμφωνα με μια έρευνα γνώμης μεγάλων αμερικανικών εταιρειών, μόνο το 28% από αυτές μπορούσε να συσχετίσει τα μέτρα ικανοποίησης του πελάτη με τα οικονομικά αποτελέσματα και μόνο το 27% με τις αποδόσεις των μετοχών [19]. Σύμφωνα με τους Anderson et al. [2], η αποτυχία στην σύνδεση μεταξύ ικανοποίησης του πελάτη και οικονομικής απόδοσης μπορεί να αποθαρρύνει τις εταιρείες από την επένδυση στην ποιότητα του προϊόντος και την ικανοποίηση πελατών.

Μια μέθοδος για την αξιολόγηση της επίδρασης της ικανοποίησης στη διατήρηση των πελατών και του μεριδίου αγοράς παρουσιάστηκε για πρώτη φορά στην εργασία των Rust and Zahorik [36]. Μια σχετική μελέτη από τους Anderson and Sullivan [1] εντόπισε παράγοντες που επηρεάζουν την ικανοποίηση του πελάτη, μελέτησε τη συμβολή της ικανοποίησης του πελάτη στην οικονομική απόδοση μιας εταιρείας και παρατήρησε ασυμμετρίες στις επιπτώσεις αντίθετων επιδράσεων (ικανοποίηση, δυσαρέσκεια) στο κέρδος, γεγονός που συμφωνεί με τις εκτιμήσεις των ειδικών και τις εμπειρικές μελέτες που υποδηλώνουν ότι η προσέλκυση νέων πελατών είναι 3 έως 30 φορές πιο κοστοβόρα από την διατήρηση των υπαρχόντων πελατών. Σε παρόμοια αποτελέσματα οδηγεί η έρευνα που αφορά τον αλληλεπίδραση μεταξύ ικανοποίησης του πελάτη και στρατηγικής μάρκετινγκ, όπου υποδεικνύεται ότι η διατήρηση ενός υπάρχοντος πελάτη μπορεί να είναι καλύτερη στρατηγική από την προσέλκυση ενός νέου πελάτη (βλ. για παράδειγμα τις εργασίες των Fornell and Wernerfelt [9], [10]).

Παρόλο που πολλές μελέτες έχουν εξετάσει ορισμένους παράγοντες οι οποίοι συνδέουν την ικανοποίηση πελατών με τα οικονομικά αποτελέσματα των επιχειρήσεων, δεν έχει προταθεί καμία ολιστική προσέγγιση, κυρίως λόγω πολλών άλλων παραγόντων που μπορεί να επηρεάσουν σημαντικά αυτήν τη σύνδεση (π.χ., χαρακτηριστικά προϊόντος ή της εταιρείας, αγορά ή άλλοι περιβαλλοντικοί παράγοντες). Ωστόσο, είναι ευρέως αποδεκτό ότι μεγαλύτερη ικανοποίηση πελατών οδηγεί σε υψηλότερα επίπεδα πρόθεσης επαναλήψεων αγορών και, στη συνέχεια, σε υψηλότερα επίπεδα εσόδων και κερδοφορίας (δείτε την εργασία των Grigoroudis and Siskos [14] για μια λεπτομερή συζήτηση). Επιπλέον, άλλοι ενδιάμεσοι παράγοντες (π.χ. διατήρηση πελατών, πιστότητα πελατών) μπορεί να αυξήσουν την πολυπλοκότητα της προηγούμενης σύνδεσης.

Για τα συστήματα παραγωγής, η ικανοποίηση πελατών είναι μια σύνθετη έννοια που εξαρτάται από πολλούς παράγοντες, όπως η ποιότητα του προϊόντος, η τιμή, οι όροι εγγύησης και ο χρόνος παράδοσης των παραγγελιών στους πελάτες. Παρόλο που η ικανοποίηση πελατών έχει λάβει σημαντική προσοχή από τους ερευνητές στον τομέα του μάρκετινγκ, δεν έχει μελετηθεί ακόμα στον ίδιο βαθμό σε συνδυασμό με τα προβλήματα ελέγχου παραγωγής και ποιότητας. Η πλειοψηφία των μελετών που αφορούν την ικανοποίηση πελατών επικεντρώνονται στην τιμολόγηση των προϊόντων και στις αποφάσεις για τον χρόνο παράδοσης, ακόμα και αν η ποιότητα του προϊόντος είναι ένας καίριος παράγοντας της ικανοποίησης του πελάτη. Στην εργασία των Deng et al. [8], προτείνονται μοντέλα για τη μελέτη της συμπεριφοράς των πελατών σε πολυπεριοδικά σύστημα αποθεμάτων με μερικώς παρατηρήσιμη, εξαρτώμενη από το επίπεδο εξυπηρέτησης ζήτηση, όπου δεν είναι επιθυμητή η έλλειψη αποθεμάτων. Μια κάπως δυαδική κατάσταση εξετάζεται στη μελέτη των Liberopoulos et al. [28], όπου ένας μόνος πελάτης επιλέγει τυχαία μεταξύ δύο προμηθευτών ανάλογα με παράγοντες αξιοπιστίας, οι οποίοι είναι φθίνουσες συναρτήσεις των ελλείψεων αποθεμάτων που έχουν παρουσιαστεί στο παρελθόν κατά την παραγγελία από κάθε εταιρεία. Οι εταιρείες λαμβάνουν δυναμικές αποφάσεις αποθεμάτων παρατηρώντας τα τρέχοντα αποθέματα και τα επίπεδα αξιοπιστίας των δύο ανταγωνιζόμενων εταιρειών. Οι Kostami and Rajagopalan [25], εξέτασαν τις αλληλεπιδράσεις μεταξύ ποιότητας, ταχύτητας και τιμής σε ένα μονοπώλιο, χρησιμοποιώντας εργαλεία από την θεωρία ουρών και την κυρτή βελτιστοποίηση. Πρόσφατα οι Liberopoulos and Deligiannis [29] μελέτησαν ένα πρόβλημα αποθεμάτων πολλών περιόδων, στο οποίο ένας προμηθευτής πωλεί ένα προϊόν σε έναν αγοραστή. Ο αγοραστής αξιολογεί τον προμηθευτή με βάση το ιστορικό εξυπηρέτησης και θέτει ανεξάρτητες και όμοια κατανεμημένες τυχαίες παραγγελίες σε κάθε περίοδο, των οποίων η κατανομή πιθανοτήτων εξαρτάται από την αξιολόγηση του προμηθευτή.

Στη βιβλιογραφία, η ποιότητα καθορίζεται συνήθως από τις αποκλίσεις των κύριων λειτουργικών χαρακτηριστικών ενός προϊόντος από την καθορισμένη τιμή-στόχο της προδιαγραφής σχεδιασμού του προϊόντος. Οι οικονομικές απώλειες που προκαλούνται από αυτές τις αποκλίσεις ονομάζονται απώλειες ποιότητας. Αυτά τα κόστη μπορεί να περιλαμβάνουν απώλεια πωλήσεων, απώλεια κύρους του παραγωγού, κλπ. Μια συνηθισμένη συνάρτηση απώλειας ποιότητας είναι η τετραγωνική συνάρτηση, η οποία έχει προταθεί από τον Taguchi et al. [39]. Αυτή η προσέγγιση είναι μια αρκετά απλή προσέγγιση και δεν περιγράφει την πολύπλοκη δυναμική μεταξύ των αποφάσεων παραγωγής, ποιότητας προϊόντος, ικανοποίησης πελατών και μεριδίων αγοράς.

Πρόσφατα υπήρξε αυξανόμενο ενδιαφέρον για τη σύνδεση του ελέγχου παραγωγής και του ελέγχου ποιότητας. Προς αυτή την κατεύθυνση, προβλήματα στα οποία οι αποφάσεις σχεδιασμού ή ελέγχου παραγωγής συνδέονται με στρατηγικές ελέγχου ποιότητας έχουν λάβει σημαντική προσοχή [46], [20], [11], [27], [7], [34]. Μια πρόσφατη βιβλιογραφική ανασκόπηση σε αυτό το θέμα μπορεί να βρεθεί στην εργασία [17].

Μια συνηθισμένη πρακτική ελέγχου ποιότητας είναι ο σχεδιασμός πλήρων σχεδίων ελέγχου, επίσης γνωστού ως εξαντλητικού ελέγχου ποιότητας. Λεπτομερείς βιβλιογραφικές ανασκοπήσεις σχετικά με τον σχεδιασμό του εξαντλητικού ελέγχου μπορούν να βρεθούν στα άρθρα [41] και [38]. Το πρόβλημα του συντονισμού παραγωγής και ελέγχου ποιότητας σε συστήματα παραγωγής όταν εφαρμόζεται εξεντλητικός έλεγχος ποιότητας έχει επίσης μελετηθεί λεπτομερώς στις εργασίες [26], [18], [15], [16]. Σε αυτά τα άρθρα φαίνεται καθαρά ότι η από κοινού εξέταση των προβλημάτων παραγωγής και ελέγχου ποιότητας οδηγεί σε σημαντική βελτίωση της οικονομικής απόδοσης των συστημάτων κατασκευής.

Ο στόχος μας σε αυτή την εργασία είναι να εξετάσουμε πώς οι αποφάσεις σχετικά με τις προδιαγραφές ποιότητας των παραγόμενων προϊόντων αλληλεπιδρούν με την ικανοποίηση πελατών και το μερίδιο αγοράς, έτσι ώστε να προτείνουμε απλές και αποτελεσματικές πολιτικές, με σκοπό την αύξηση της κερδοφορίας των συστημάτων παραγωγής.

1.3 Δομή της διατριβής, μεθοδολογία, αποτελέσματα

Στόχος της εργασίας είναι η αριθμητική διερεύνηση της βέλτιστης πολιτικής παραγωγής και ελέγχου ποιότητας σε μια σειρά από συστήματα παραγωγής όταν ο μέσος ρυθμός πωλήσεων καθορίζεται από την ικανοποίηση των πελατών σχετικά με την ποιότητα των παραγόμενων προϊόντων. Σε κάποιες περιπτώσεις προτείνουμε απλές ευρετικές πολιτικές που αποδεικνύονται αποτελεσματικά υποκατάστατα των βέλτιστων πολιτικών.

Για τη μαθηματική περιγραφή των εξεταζόμενων προβλημάτων χρησιμοποιήθηκε η θεωρία στοχαστικού δυναμικού προγραμματισμού που είναι κατάλληλη για την μοντελοποίηση στοχαστικών προβλημάτων βελτίστου ελέγχου. ουρών ή αναμονής. Μερικές επιτυχημένες εφαρμογές στοχαστικού δυναμικού προγραμματισμού σε συστήματα παραγωγής μπορεί να δει στις εργασίες [42], [43], [44], [45], [4], [40]. Για την ανάλυση των ευρετικών πολιτικών χρησιμοποιήθηκε η θεωρία ουρών ή συστημάτων αναμονής. Πρώτος χρησιμοποίησε τη θεωρία ουρών σε προβλήματα συστημάτων παραγωγής και αποθεματικών συστημάτων ο Morse [31]. Έκτοτε έχει χρησιμοποιηθεί κατά κόρον σε προβλήματα παραγωγής με σημαντική επιτυχία. Εκτενείς βιβλιογραφικές επισκοπήσεις των εφαρμογών της θεωρίας ουρών στα συστήματα παραγωγής παρουσιάζεται από τους Govil and Fu Kelly [13], Papadopoulos et al. [33]. Ένας λόγος που προτιμήθηκε το συγκεκριμένο πλαίσιο μαθηματικής μοντελοποίησης είναι ότι περιγράφει με ξεκάθαρο και ακριβή τρόπο τη λειτουργία συστημάτων όπως αυτά που εξετάζονται εδώ.

Η παρούσα διατριβή αποτελείται από τρία επιπλέον κεφάλαια:

Στο Κεφάλαιο 2, περιγράφουμε ένα σύνθετο δίκτυο παραγωγής κατά παραγγελίες (make-to-order) με συμπεριφορές πελατών εξαρτώμενες από το επίπεδο ποιότητας των τελικών προϊόντων. Διατυπώνουμε ένα πρόβλημα ελέγχου ποιότητας για τη μεγιστοποίηση του μέσου ρυθμού κέρδους του συστήματος. Προκύπτει ότι η βέλτιστη πολιτική απαιτεί ακριβή γνώση της τρέχουσας κατάστασης της αγοράς (αριθμός των τακτικών και περιστασιακών πελατών) και του πλήθους εκκρεμών παραγγελιών. Στην συνέχεια, προτείνουμε μια πολιτική ελέγχου ποιότητας τύπου κατωφλίου, η οποία είναι εύκολα εφαρμόσιμη και δεν απαιτεί καμία πληροφορία κατάστασης. Τέλος παρουσιάζουμε μια αριθμητική διερεύνηση μεταξύ των δύο πολιτικών (βέλτιστη και ευρετική).

Στο Κεφάλαιο 3, εξετάζουμε την περίπτωση όπου το επίπεδο ικανοποίησης των πελατών μετά την αγορά ενός προϊόντος εξαρτάται από την ποιότητα του προϊόντος καθώς και από το επίπεδο ικανοποίησης των πελατών πριν από την αγορά. Συνεπώς οι πελάτες έχουν μνήμη του προηγούμενου επιπέδου ικανοποίησης και αυτό επηρεάζει και την τρέχουσα ικανοποίηση. Αρχικά, περιγράφουμε ένα σύστημα παραγωγής ενός σταδίου που παράγει προς κατά παραγγελίες (make-to-order) του οποίου η αγορά αποτελείται από τακτικούς και περιστασιακούς πελάτες. Ακολούθως, διερευνούμε μέσω αριθμητικών πειραμάτων την βέλτιστη πολιτική ελέγχου ποιότητας που μεγιστοποιεί τον μέσο ρυθμό κέρδους του συστήματος βάσει ακριβούς γνώσης της τρέχουσας κατάστασης. Ακολούθως προτείνουμε μια ευρετική πολιτική ελέγχου ποιότητας τύπου κατωφλίου, η οποία είναι εύχρηστη, υπολογιστικά αποδοτική, ενώ δεν απαιτεί καμία πληροφορία κατάστασης. Τέλος πραγματοποιήσαμε μια αριθμητική διερεύνηση μεταξύ της ευρετικής πολιτικής και της βέλτιστης. Τα αποτελέσματα της αριθμητικής διερεύνησης έδειξαν ότι η προτεινόμενη ευρετική πολιτικήςείναι μια καλή εναλλακτική πρόταση.

Στο κεφάλαιο 4, μελετούμε την αλληλεπίδρασης της ποιότητας και του ελέγχου παραγωγής, λαμβάνοντας υπόψη τις επιπτώσεις αυτών των αποφάσεων στο μέγεθος της αγοράς μιας εταιρείας σε συστήματα παραγωγής προς αποθεματοποίηση (make-to-stock). Περιγράφουμε ένα σύστημα ενός σταδίου, που ακολουθεί την προσέγγιση παραγωγής προς αποθεματοποίηση, του οποίου η αγορά αποτελείται από τακτικούς και περιστασιακούς πελάτες. Στη συνέχεια, διερευνούμε μέσω αριθμητικών πειραμάτων την βέλτιστη πολιτική παραγωγής και ελέγχου ποιότητας που μεγιστοποιεί τον μέσο ρυθμό κέρδους του συστήματος συναρτήσει της τρέχουσας κατάστασης.

Τέλος στο Κεφάλαιο 5, παρουσιάζουμε τα συμπεράσματα και κατευθύνσεις για μελλοντική έρευνα.

2 Σχέση ποιότητας προϊόντος και κερδοφορίας σε σύνθετα δίκτυα παραγωγής κατά παραγγελίες όταν η αγοραστική συμπεριφορά των καταναλωτών δεν έχει μνήμη

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο παρουσιάζεται ένα μοντέλο συστημάτων αναμονής που αναπτύχθηκε για την κατανόηση του τρόπου με τον οποίο η ποιότητα του προϊόντος μπορεί να επηρεάσει την κερδοφορία των συστημάτων παραγωγής, όταν οι καταναλωτές βασίζουν τη μελλοντική τάση ζήτησης στην ποιότητα του προϊόντος που αγόρασαν πιο πρόσφατα. Η εξάρτηση της μελλοντικής ζήτησης ενός πελάτη μόνο από την πρόσφατη εμπειρία του συνιστά έλλειψη μνήμης και αποτελεί μία απλουστευτική παραδοχή η οποία επιτρέπει την προσεγγιστική ανάλυση συστημάτων με πολλές μηχανές και ελεύθερη χωροθέτηση χρησιμοποιώντας μοντέλα δικτύων αναμονής Jackson. Τέτοια συστήματα είναι γνωστά και ως καταστήματα εργασιών (jobshops).

Εξετάζουμε ένα δίκτυο παραγωγής, που δέχεται παραγγελίες από τακτικούς και περιστασιακούς πελάτες, με τους πρώτους να έχουν υψηλότερο μέσο ρυθμό ζήτησης. Το σύστημα δεν διατηρεί έτοιμο απόθεμα και λειτουργεί μόνον όταν εκκρεμούν παραγγελίες. Κάθε παραγόμενο προϊόν ανήκει σε κάποια κατηγορία ποιότητας από ένα διακριτό σύνολο (π.χ. {άριστο, μέτριο, κ.λπ}). Η πιθανότητα κάθε κατηγορίας θεωρείται γνωστή και σταθερή. Το προϊόν υπόκειται σε έλεγχο ποιότητας που εντοπίζει την ακριβή κατηγορία (χωρίς σφάλμα) και αποφασίζεται εάν θα απορριφθεί ως μη συμμορφούμενο ή θα αποσταλεί σε πελάτη. Στην τελευταία περίπτωση, ο πελάτης που αγοράζει το προϊόν είτε θα ικανοποιηθεί είτε όχι, αναλόγως του επιπέδου ποιότητας και εντάσσεται στην ομάδα των τακτικών πελατών ή στην ομάδα των περιστασιακών πελατών με συμπληρωματικές πιθανότητες, οι οποίες εξαρτώνται από το επίπεδο ποιότητας του προϊόντος που μόλις προμηθεύτηκε.

Η κερδοφορία του συστήματος εξαρτάται από τις αποφάσεις απόρριψης και αποδοχής του ελέγχου ποιότητας. Όταν ο έλεγχος είναι αυστηρός τότε απορρίπτονται πολλά προϊόντα με συνεπαγόμενο κόστος (επανάληψης της παραγωγής και προμήθειας νέων πρώτων υλών) αλλά τότε τα προϊόντα είναι υψηλής ποιότητας, η πιθανότητα ικανοποίησης πελατών υψηλή και η ζήτηση μεγάλη. Όταν ο έλεγχος είναι χαλαρός συμβαίνουν τα αντίθετα. Στόχος είναι η εύρεση της βέλτιστης πολιτικής κατάταξης των προϊόντων σε απορριπτέα ή αποδεκτά.

Η λύση του προβλήματος γίνεται αρχικά υπολογιστικά με δυναμικό προγραμματισμό. Τα αποτελέσματα των αριθμητικών πειραμάτων, που πραγματοποιήθηκαν με δυναμικό προγραμματισμό, δείχνουν ότι η βέλτιστη πολιτική εξαρτάται από την κατάσταση του συστήματος, είναι πολύπλοκη και έχει αυξημένες υπολογιστικές απαιτήσεις.

Στη συνέχεια προτείνεται μια απλή ευρετική πολιτική τύπου κατωφλίου, για την ανάλυση και την βελτιστοποίηση της οποίας χρησιμοποιούνται μοντέλα κλειστών δικτύων αναμονής με ελάχιστες υπολογιστικές απαιτήσεις. Τα αριθμητικοί αποτελέσματα υποδεικνύουν ότι η προτεινόμενη ευρετική πολιτική αποδίδει σχεδόν το ίδιο καλά με τη βέλτιστη πολιτική.

Η εφαρμογή μοντέλων συστημάτων αναμονής στην τμηματοποίηση της αγοράς αποτελεί μία σημαντική συνεισφορά της παρούσας διατριβής, επειδή συνδέει *αμφίδρομα* την ποιότητα, παραγωγικότητα και κερδοφορία ενός συστήματος παραγωγής με την καταναλωτική συμπεριφορά και το μάρκετινγκ.

Το κεφάλαιο είναι διαρθρωμένο ως εξής. Στην Παράγραφο 2.2, περιγράφουμε ένα σύνθετο δίκτυο παραγωγής κατά παραγγελία με συμπεριφορές πελατών εξαρτώμενες από το επίπεδο ποιότητας των τελικών προϊόντων. Διατυπώνουμε ένα πρόβλημα ελέγχου ποιότητας για τη μεγιστοποίηση του μέσου ρυθμού κέρδους του συστήματος. Προκύπτει ότι η βέλτιστη πολιτική απαιτεί ακριβή γνώση της τρέχουσας κατάστασης της αγοράς (αριθμός των τακτικών και περιστασιακών πελατών) και του πλήθους εκκρεμών παραγγελιών. Στην Παράγραφο 2.3, προτείνουμε μια πολιτική ελέγχου ποιότητας τύπου κατωφλίου, η οποία είναι εύκολα εφαρμόσιμη και δεν απαιτεί καμία πληροφορία κατάστασης. Στην Παράγραφο 2.4 παρουσιάζουμε μια αριθμητική διερεύνηση μεταξύ των δύο πολιτικών (βέλτιστη και ευρετική). Τα συμπεράσματα παρουσιάζονται στην Παράγραφο 2.5.

2.2 Περιγραφή συστήματος, μοντελοποίηση και λήψη βέλτιστων αποφάσεων

Εξετάζουμε ένα σύστημα παραγωγής κατά παραγγελίες (make-to-order) που κατασκευάζει έναν μοναδικό τύπο προϊόντος. Το σύστημα εξυπηρετεί μια αγορά που αποτελείται από συνολικά M πελάτες, όπου το M είναι σταθερό. Η αγορά διαιρείται σε δύο διακριτές κατηγορίες: η κατηγορία των τακτικών ή ικανοποιημένων πελατών, συμβολίζεται με $i = 1$, και η κατηγορία των περιστασιακών ή αδιαφορών πελατών, $i = 2$. Το σύστημα λαμβάνει παραγγελίες από πελάτες οποιασδήποτε κατηγορίας που επί του παρόντος δεν έχουν καμία εκκρεμή παραγγελία. Έτσι, το M είναι το άθροισμα τριών μεταβαλλόμενων με τον χρόνο μεταβλητών κατάστασης:

n_1 πλήθος τακτικών πελατών που δεν έχουν εκκρεμείς παραγγελίες στο σύστημα,

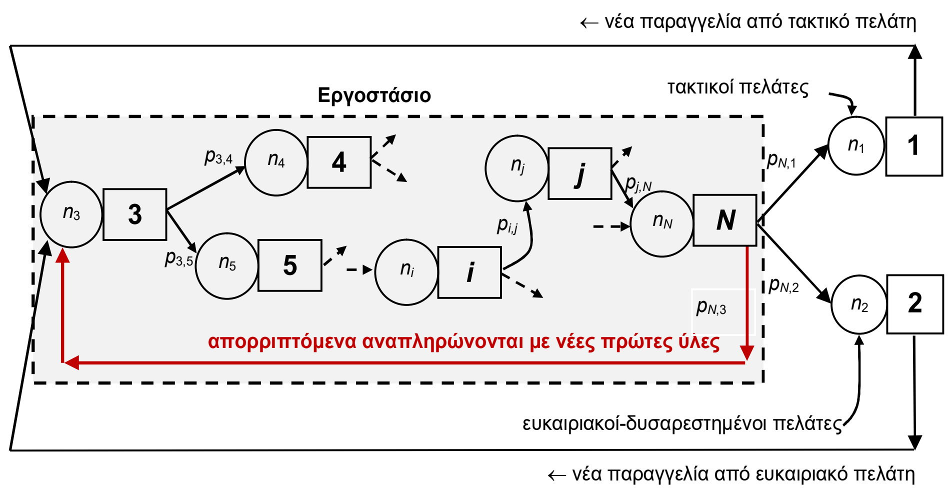
n_2 πλήθος περιστασιακών πελατών που δεν έχουν εκκρεμείς παραγγελίες στο σύστημα

$M - n_1 - n_2$ πλήθος πελατών που έχουν εκκρεμείς παραγγελίες στο σύστημα παραγωγής.

Οι παραγγελίες των πελατών είναι τυχαίες και ακολουθούν την κατανομή Poisson. Κάθε πελάτης ζητά μια μονάδα προϊόντος και ο μέσος ρυθμός ζήτησης εξαρτάται από την κατηγορία του πελάτη, λ_i, $i = 1, 2$, όπου $\lambda_1 > \lambda_2$. Έτσι, ο μέσος ρυθμός ζήτησης στην κατάσταση (n_1, n_2) είναι $\lambda_1 n_1 + \lambda_2 n_2$. Όταν ένας πελάτης κάνει μια παραγγελία, μια μονάδα πρώτης ύλης εισέρχεται στο πρώτο στάδιο παραγωγής του συστήματος, και όταν ένα τελικό προϊόν ολοκληρώνεται, η ποιότητά του ελέγχεται και λαμβάνεται απόφαση εάν αυτό το προϊόν θα απορριφθεί ή θα αποσταλεί για να καλύψει μια εκκρεμή παραγγελία. Κάθε εξερχόμενο προϊόν έχει αποκτήσει ένα συγκεκριμένο επίπεδο ποιότητας q με αντίστοιχη πιθανότητα p_q, όπου $q = 0, 1, ..., L$, $p_0 + ... + p_L = 1$, και η ποιότητα του προϊόντος χειροτερεύει με την αύξηση του q. Εάν ένα προϊόν με επίπεδο ποιότητας q πωλείται σε έναν πελάτη, τότε αυτός ο πελάτης θα είναι τελικά ικανοποιημένος με πιθανότητα s_q ή μη ικανοποιημένος με τη συμπληρωματική πιθανότητα. Οι ικανοποιημένοι πελάτες εντάσσονται στην τακτική κατηγορία ενώ οι μη ικανοποιημένοι γίνονται περιστασιακοί πελάτες. Τα προϊόντα υψηλής ποιότητας έχουν υψηλότερες πιθανότητες ικανοποίησης. Έτσι, $s_0 > s_1 > \ldots > s_L$.

Το σύστημα παραγωγής μοντελοποιείται ως ένα κλειστό δίκτυο αναμονής Jackson, όπου ο κάθε κόμβος αποτελείται από μια μηχανή. Αυτή η υπόθεση υιοθετείται εδώ για λόγους απλότητας. Η ανάλυση που ακολουθεί μπορεί να επεκταθεί σε πιο γενικά Μαρκοβιανά δίκτυα παραγωγής.

Έχουμε συμβολίσει i = 1, 2 τις αγορές τακτικών και περιστασιακών πελατών αντίστοίχως. Για να ενοποιήσουμε τον συμβολισμό κυρίως της παραγράφου που ακολουθεί την παρούσα, υποθέτουμε ότι το σύστημα διαθέτει μια μονάδα παραγωγής με $N - 2$ μηχανές που συμβολίζονται ως $i = 3, 4, \ldots, N$, όπου η μηχανή 3 επεξεργάζεται τις πρώτες ύλες που αντιστοιχούν σε κάθε νέα παραγγελία και η μηχανή N παράγει το τελικό προϊόν, όπως φαίνεται στο Σχήμα 2.1. Κάθε μηχανή τροφοδοτείται από έναν απεριόριστης χωρητικότητας ενδιάμεσο αποθηκευτικό χώρο, στον οποίο αποθηκεύονται προσωρινά τα ενδιάμεσα προϊόντα που προωθούνται από άλλες μηχανές. Οι χρόνοι επεξεργασίας στη μηχανή i είναι ανεξάρτητες τυχαίες μεταβλητές εκθετικά κατανεμημένες με μέση τιμή $1/\mu_i$.



Σχήμα 2.1 Σύστημα παραγωγής κατά παραγγελία με δύο κατηγορίες πελατών και απόρριψη ελαττωματικών προϊόντων.

Η ροή των προϊόντων στο σύστημα περιγράφεται από ένα πίνακα $\Pi = [p_{i,j}]$, όπου $p_{i,j}$ είναι η πιθανότητα δρομολόγησης από τη μηχανή i στη μηχανή j. Για παράδειγμα, αν οι μηχανές i και i+1 είναι διαδοχικές σε μια γραμμή παραγωγής τότε έχουμε ότι $p_{i,i+1}$ = 1, ενώ για πιο γενικές γεωμετρίες ισχύει $p_{i,j} \geq 0$ και $\sum_j p_{i,j} = 1$ για κάθε i. Αν ένα αντικείμενο ποιότητας q παραχθεί από τη μηχανή N και πωληθεί σε έναν πελάτη, τότε είτε αυτός ο πελάτης θα είναι ικανοποιημένος και θα ενταχθεί στην ομάδα των πελατών κλάσης 1 με αντίστοιχη πιθανότητα δρομολόγησης $p_{N,1} = s_q$, είτε θα είναι ανικανοποίητος και θα γίνει πελάτης κλάσης 2 με αντίστοιχη πιθανότητα δρομολόγησης $p_{N,2} = 1 - s_q$.

Ο στόχος είναι να βρεθεί η βέλτιστη πολιτική ελέγχου, η οποία καθορίζει εάν θα πωληθεί ή θα απορριφθεί ένα περαγόμενο προϊόν συγκεκριμένου επιπέδου ποιότητας q, ώστε να μεγιστοποιηθεί ο μέσος ρυθμός κέρδους του συστήματος. Ο μέσος ρυθμός κέρδους εξαρτάται από το ρυθμό πώλησης (παραγωγικότητα του συστήματος), το κόστος των απορριφθέντων αντικειμένων και τα κόστη αποθεματοποίησης και εκκρεμών παραγγελιών με τις αντίστοιχες παραμέτρους.

r κέρδος από την πώληση μιας μονάδας προϊόντος,

c μοναδιαίο κόστος απόρριψης (κόστος από την απόρριψη ή/και επανακατεργασία ενός προϊόντος),

b_i μοναδιαίο κόστος αποθέματος και εκκρεμών παραγγελιών στον κόμβο i, $i = 3, \ldots, N$.

Οι παράμετροι b_i περιλαμβάνουν τα μοναδιαία αποθεματικά κόστη (κόστη διατήρησης στο απόθεμα μιας μονάδας προϊόντος για μια χρονική μονάδα) και τα μοναδιαία κόστη εκκρεμών παραγγελιών (κόστος καθυστέρησης μιας παραγγελιάς για μια μονάδα του χρόνου) του κόμβου i, $i = 3, 4, \ldots, N$.

Η κατάσταση του συστήματος περιγράφεται από το διάνυσμα $n = (n_1, n_2, n_3, \ldots, n_N)$ όπου ο χώρος καταστάσεων είναι $Z = \{n \mid n_1 + n_2 + n_3 + \ldots + n_N = M$ και $n_i \geq 0$ για όλα τα $i = 1, \ldots, N\}$. Κάθε φορά που η μηχανή N παράγει ένα τελικό προϊόν, παρατηρούμε το επίπεδο ποιότητας του προϊόντος και την κατάσταση του συστήματος n και παίρνουμε μια απόφαση $\pi(n, q)$, όπου $\pi(n, q) = 1$ αν το προϊόν προωθείται σε κάποιον πελάτη ενώ $\pi(n, q) = 0$ διαφορετικά. Όταν ένα προϊόν απορρίπτεται ως ακατάλληλο μια μονάδα πρώτης ύλης προωθείται στην αποθήκη του κόμβου 3, συνεπώς η μεταβλητή κατάστασης n_N μειώνεται κατά μια μονάδα, ενώ η n_3 αυξάνει κατά μια μονάδα. Όταν ένα προϊόν πωλείται η μεταβλητή, n_N επίσης μειώνεται κατά ένα και είτε αυξάνει κατά μια μονάδα η n_1, όταν ο πελάτης ικανοποιείται από την ποιότητα του προϊόντος, είτε αυξάνει το n_2 στην αντίθετη περίπτωση. Ανάλογες αλλαγές στην κατάσταση του συστήματος συμβαίνουν όταν μια οντότητα (ένα προϊόν ή μια παραγγελία) κινείται από έναν κόμβο του συστήματος σε έναν άλλο: η κίνηση από τον i στον j οδηγεί σε μείωση της n_i κατά μια μονάδα και στην αύξηση της n_j κατά μια μονάδα. Η νέα κατάσταση εκφράζεται σε διανυσματική μορφή ως $n - e_i + e_j$ όπου e_i είναι ένα διάνυσμα που περιέχει την μονάδα στην i–στη θέση και 0 στις άλλες θέσεις.

Ορίζονται τρεις τύποι γεγονότων:

- άφιξη πελάτη τύπου i (i = 1 για τακτικούς πελάτες, i = 2 για περιστασιακούς πελάτες),
- ολοκλήρωση ενός ημιτελούς προϊόντος από την μηχανή $i = 3, \ldots, N - 1$,
- παραγωγή ενός τελικού προϊόντος ποιότητας q από την μηχανή N και λήψη απόφασης πώλησης ή απόρριψης του προϊόντος.

Επειδή οι χρόνοι μεταξύ των διαφόρων γεγονότων είναι τυχαίοι και εκθετικά κατανεμημένοι, μπορούμε να περιγράψουμε μαθηματικά το πρόβλημα βελτιστοποίησης ώς μια Μαρκοβιανή διαδικασία αποφάσεων. Αυτό επιτυγχάνετα χρησιμοποιώντας την τεχνική της ομοιομορφοποίησης (βλ. [30], [35], [3] [37]) και μια ομογενής διαδικασία Poissson με ρυθμό $\nu = M\lambda_1 + \mu_3 + \mu_4 + \ldots + \mu_N$. Έστω $V_k(n)$ η τιμή του μέγιστου αναμενόμενου κέρδους των πρώτων k γεγονότων της διαδικασίας Poisson, όταν η αρχική κατάσταση είναι n. Η εξίσωση δυναμικού προγραμματισμού για το βέλτιστο αναμενόμενο κέρδος των πρώτων k + 1 γεγονότων είναι

$$\begin{aligned} V_{k+1}(n) = \frac{1}{\nu} \cdot \Bigg\{ & n_1 \lambda_1 V_k(n - e_1 + e_3) + n_2 \lambda_2 V_k(n - e_2 + e_3) \\ & + \sum_{i=3}^{N-1} \left[\mu_i I_{n_i > 0} \sum_{j=3}^{N} p_{i,j} V_k(n - e_i + e_j) \right] \\ & + \mu_N I_{n_N > 0} \sum_{q=0}^{L} p_q \max \big[s_q V_k(n - e_N + e_1) + (1 - s_q) V_k(n - e_N + e_2) + r, \\ & \qquad V_k(n - e_N + e_3) - c \big] \\ & + \left(\nu - n_1 \lambda_1 - n_2 \lambda_2 - \sum_{i=3}^{N} \mu_i I_{n_i > 0} \right) V_k(n) - \sum_{i=3}^{N} b_i n_i \Bigg\} \end{aligned} \tag{2.1}$$

για κάθε $n \in Z$, όπου $V_0(n) \equiv 0$ και

$$I_C = \begin{cases} 1, & \text{ικανοποιείται η } C \\ 0, & \text{διαφορετικά} \end{cases}$$

Οι όροι μέσα στις αγκύλες στην Εξίσωση 2.1 αντιστοιχούν στην άφιξη ενός τακτικού ή περιστασιακού πελάτη (πρώτοι δύο όροι), ολοκλήρωση ενός ημιτελούς εξαρτήματος (οι δύο ενσωματωμένοι αθροιστικοί όροι), παραγωγή ενός τελικού προϊόντος ακολουθούμενη από έλεγχο και την πιο κερδοφόρα απόφαση (τέταρτος όρος με τον τελεστή μεγιστοποίσης), μια αυτομετάβαση (ψευδογεγονός, πέμπτος όρος) και, τέλος, το συνολικό κόστος παραμονής στην κατάσταση n.

Επειδή όλες οι καταστάσεις έχουν αυτομεταβάσεις (αφού, ο ρυθμός ν είναι μεγαλύτερος από τον συνολικό ρυθμό μετάβασης από οποιαδήποτε κατάσταση), είναι απεριοδικές. Επίσης, το πλήθος των καταστάσεων στο Z και ο αριθμός των αποφάσεων σε κάθε κατάσταση είναι πεπερασμένα. Ως αποτέλεσμα ([35] Θεώρημα 9.4.5), το βέλτιστο μακροπρόθεσμο μέσο κέρδος J^* του συστήματος δίνεται από την μέθοδο διαδοχικών προσεγγίσεων όπως παρουσιάζεται στην ακόλουθη

Πρόταση 2.1: Υποθέτουμε ότι κάθε βέλτιστη αναμενόμενη στάσιμη ντετερμινιστική πολιτική έχει έναν απεριοδικό πίνακα μεταβάσεων, τότε ο ρυθμός μέσου μακροχρόνιου κέρδους J^* δίνεται από

$$J^* = \nu \lim_{k \to \infty} \left[V_k(n) - V_{k-1}(n) \right] \tag{2.2}$$

για κάθε $n \in Z$ και για κάθε $V_0(n)$.

Μια κατάσταση λέγεται *απεριοδική* αν οι επιστροφές σε αυτήν την κατάσταση μπορούν να συμβούν άτακτα, μετά από έναν μη περιοδικό αριθμό μεταβάσεων.

Για παράδειγμα, ένα σύστημα το οποίο επισκέπτεται δύο μόνο καταστάσεις *εναλλάξ* (π.χ. $1 \to 2 \to 1 \to \ldots$) είναι περιοδικό με περίοδο 2.

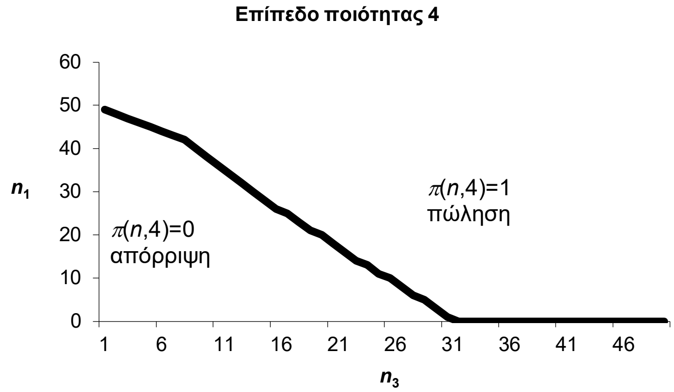
Στο σύστημά μας, ο πίνακας μεταβάσεων κάθε πολιτικής είναι απεριοδικός επειδή η πιθανότητα να παραμείνει το σύστημα στην τρέχουσα κατάσταση είναι θετική ανεξάρτητα από την απόφαση που λαμβάνεται.

Εδώ παρουσιάζουμε μια αριθμητική διερεύνηση της βέλτιστης πολιτικής ενός συστήματος με μία μηχανή (δηλαδή, N = 3), που εξυπηρετεί τακτικούς και περιστασιακούς πελάτες. Εκτελώντας αρκετές επαναλήψεις δυναμικού προγραμματισμού, υπολογίζουμε το βέλτιστο μακροπρόθεσμο μέσο κέρδος J* χρησιμοποιώντας την Εξίσωση 2.2 και τη βέλτιστη απόφαση σε κάθε κατάσταση n (ο όρος μεγιστοποίησης στην Εξίσωση 2.1). Προκύπτει ότι ακόμη και για τέτοια απλά συστήματα παραγωγής, η βέλτιστη πολιτική έχει μια αρκετά πολύπλοκη δομή.

Οι τυπικές τιμές των παραμέτρων του συστήματος που χρησιμοποιήθηκαν στα αριθμητικά πειράματα είναι $M = 50$, $\lambda_1 = 1$, $\lambda_2 = 0.1$, $\mu_3 = 60$, $r = 4$, $c = 3.5$ και $b_3 = 0.3$. Στη συνέχεια, κάνουμε μερικές επιπλέον υποθέσεις μόνο για να καθορίσουμε μερικές πιθανές τιμές για τις πιθανότητες παραγωγής σε επίπεδο ποιότητας p_q και τις αντίστοιχες πιθανότητες ικανοποίησης s_q. Η ποιότητα κάθε παραγόμενου αντικειμένου καθορίζεται από την απόλυτη απόκλιση της τιμής ενός συγκεκριμένου ποιοτικού χαρακτηριστικού (Y) από μια ιδανική τιμή ($t = 10$). Υποθέτουμε ότι η (Y) ακολουθεί μια κανονική κατανομή με μέση τιμή ίση με τη ιδανική τιμή 10 και διακύμανση ($\sigma^2 = 1$). Αυτή είναι μια συνηθισμένη υπόθεση στη βιβλιογραφία ελέγχου ποιότητας. Για κάθε εξερχόμενο αντικείμενο, το χαρακτηριστικό (Y) ελέγχεται και αντιστοιχίζεται σε μία από τις οκτώ ποιοτικές κατηγορίες ($q = 0, \ldots, 7$). Για τα πειραματικά αποτελέσματα που παρουσιάζονται εδώ, οι ποιοτικές κατηγορίες καθορίζονται ως ακολούθως. Χωρίζουμε το διάστημα $[t - 3\sigma, t + 3\sigma]$ που περιέχει τη μάζα πιθανότητας 0.997 των τιμών Y σε 16 τμήματα ίσου μήκους $3\sigma/8$, τα οποία είναι συμμετρικά ανά ζεύγη γύρω από το t. Στα συμμετρικά τμήματα $[\mathrm{t} - 3(q + 1)\sigma/8, \mathrm{t} - 3q\sigma/8]$ και $[\mathrm{t} + 3q\sigma/8, \mathrm{t} + 3(q + 1)\sigma/8]$ αποδίδεται και στα δύο το επίπεδο ποιότητας q, $q = 0, ..., 7$. Τα δύο συμμετρικά διαστήματα που αντιστοιχούν στο $q = 7$ τροποποιούνται σε $(-\infty, t - 21\sigma/8]$ και $[t + 21\sigma/8, \infty)$ ώστε να καλύπτουν όλες τις πιθανές τιμές Y. Χρησιμοποιώντας προσεγγίσεις κανονικών πιθανοτήτων, υπολογίζουμε την πιθανότητα p_q να εμπίπτει το Y σε ένα από τα δύο συμμετρικά διαστήματα που αντιστοιχούν στο επίπεδο ποιότητας q. Τέλος, υποθέτουμε ότι η πιθανότητα ικανοποίησης του πελάτη s_q είναι μια σιγμοειδής συνάρτηση του q. Οι σιγμοειδείς συναρτήσεις χρησιμοποιούνται συνήθως στη βιβλιογραφία για τη μοντελοποίηση διαφόρων σχέσεων μεταξύ της ποιότητας του προϊόντος και της ικανοποίησης του πελάτη (βλέπε, π.χ., Grigoroudis and Siskos; 2010). Οι πιθανότητες ικανοποίησης πελατών και ποιότητας προϊόντων για τις εξεταζόμενες περιπτώσεις δοκιμής παρουσιάζονται στον πίνακα 2.1.

Πίνακας 2.1 Πιθανότητες ποιότητας προϊόντος και ικανοποίησης πελατών

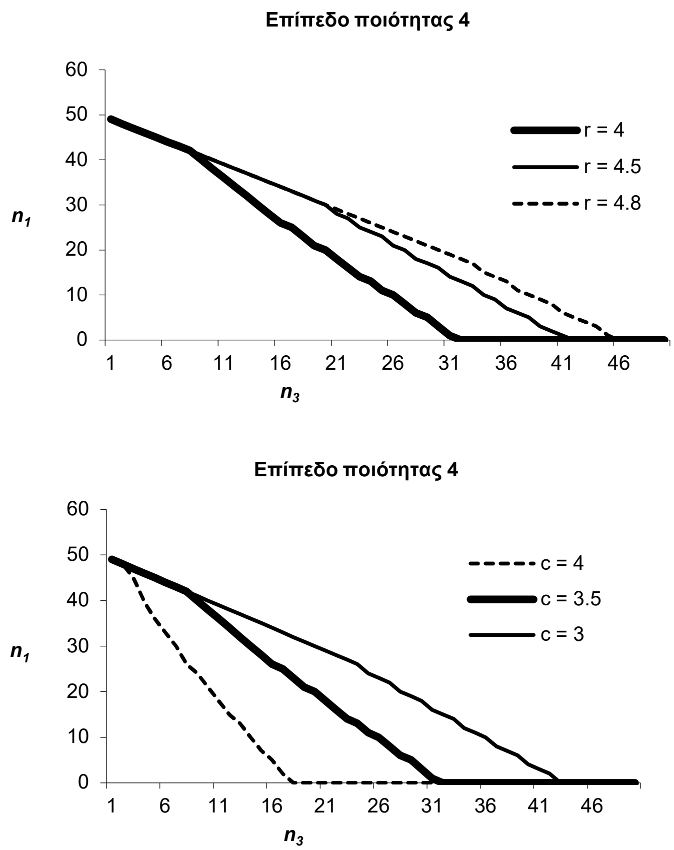
q	0	1	2	3	4	5	6	7
p_q	0.261	0.234	0.188	0.135	0.087	0.050	0.026	0.019
s_q	0.950	0.938	0.868	0.692	0.458	0.282	0.212	0.200



Σχήμα 2.2. Βέλτιστη πολιτική $\pi(n, q)$ για το επίπεδο ποιότητας $q = 4$.

Το Σχήμα 2.2 δείχνει τη βέλτιστη πολιτική ως συνάρτηση του n_1 (τακτικοί πελάτες) και του n_3 (εκκρεμείς παραγγελίες) όταν το παραγόμενο προϊόν έχει επίπεδο ποιότητας 4 (υπενθυμίζεται ότι η ποιότητα μειώνεται όσο αυξάνεται το επίπεδο q και ότι $n_2 = M - n_1 - n_3$). Για τα επίπεδα ποιότητας 0 έως 3 η βέλτιστη απόφαση είναι $\pi(n, q) = 1$ για όλες τις καταστάσεις του συστήματος n, ενώ για τα επίπεδα ποιότητας 5 έως 7 η βέλτιστη απόφαση είναι $\pi(n, q) = 0$ για όλες τις καταστάσεις. Το αποτέλεσμα αυτό συμφωνεί με τη διαίσθηση: αξίζει να απορρίπτονται προϊόντα μόνο αν η ποιότητά τους είναι κακή. Στο Σχήμα 2.2 βλέπουμε τις περιοχές των αποφάσεων αποδοχής και απόσυρσης εντός του χώρου καταστάσεων για επίπεδο ποιότητας $q = 4$. Γίνεται σαφές ότι σε αυτά τα επίπεδα ποιότητας όπου υπάρχουν δύο ανταγωνιστικές αποφάσεις, είναι βέλτιστο να πωλείται ένα προϊόν μέτριας ποιότητας (εδώ q = 4) είτε όταν το n_1 είναι μεγάλο, δηλαδή υπάρχουν ήδη πολλοί τακτικοί πελάτες στην αγορά, είτε όταν το n_3, ο αριθμός των πελατών με εκκρεμείς παραγγελίες, είναι μεγάλος και το σύστημα επιβαρύνεται με υψηλό μέσο κόστος εκκρεμών παραγγελιών b_3n_3. Για τις παραπάνω τιμές παραμέτρων, η μέθοδος διαδοχικών προσεγγίσεων εκτίμησε το βέλτιστο μακροπρόθεσμο μέσο κέρδους ως $J^* = 77.44$.

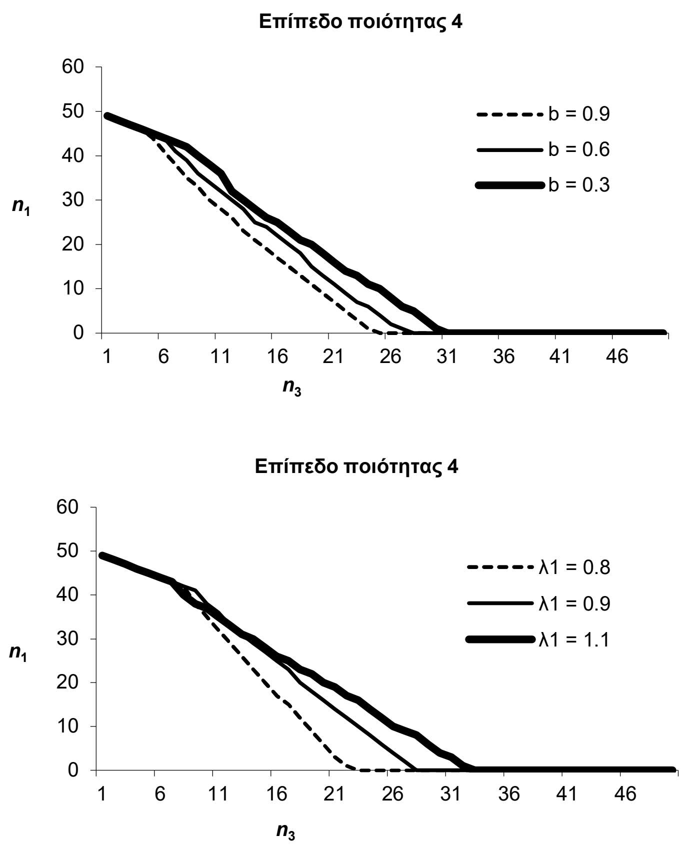
Εξετάζουμε τώρα πόσο ευαίσθητη είναι η βέλτιστη πολιτική σε αλλαγές στις τιμές των παραμέτρων. Πρώτα χρησιμοποιούμε τρεις διαφορετικές τιμές για του μοναδιαίου κέρδους r. Το αριστερό γράφημα του Σχήματος 2.3 δείχνει τις καμπύλες εναλλαγής αποφάσεων της βέλτιστης πολιτικής για το επίπεδο ποιότητας 4. Οι περιοχές στα αριστερά κάθε καμπύλης απόφασης περιλαμβάνουν όλες τις καταστάσεις (n_1, n_3) για τις οποίες μια απόφαση απόρριψης/επανακατεργασίας είναι βέλτιστη και οι περιοχές στα δεξιά αντιστοιχούν σε βέλτιστες αποφάσεις πώλησης για είδη ποιότητας επιπέδου 4. Καθώς αυξάνεται το μοναδιαίο κέρδος *r*, αυξάνεται και ο αριθμός των καταστάσεων για τις οποίες η απόρριψη είναι βέλτιστη εντός ενός συγκεκριμένου επιπέδου ποιότητας. Για το ίδιο επίπεδο ποιότητας διερευνούμε πόσο ευαίσθητη είναι η βέλτιστη πολιτική στις μεταβολές της τιμής του κόστους απόρριψης *c*. Το δεξιό γράφημα του Σχήματος 2.3 υποδηλώνει ότι όσο μειώνεται το κόστος απόρριψης η απόσυρση γίνεται πιο ελκυστική, αποτέλεσμα που συμφωνεί με τη διαίσθηση.



Σχήμα 2.3 Βέλτιστες πολιτικές για q = 4 σε σχέση με το μοναδιαίο κέρδος και το μοναδιαίο κόστος απόρριψης.

Στη συνέχεια, διερευνούμε την ευαισθησία της βέλτιστης πολιτικής στο μοναδιαίο κόστος εκκρεμών παραγγελιών b_3 και στο ρυθμό άφιξης παραγγελιών λ_1 των τακτικών πελατών. Χρησιμοποιούμε τρεις διαφορετικές τιμές για κάθε παράμετρο. Το Σχήμα 2.4 δείχνει τις βέλτιστες πολιτικές για το επίπεδο ποιότητας 4. Βλέπουμε ότι όταν το b

μειώνεται, οι αποφάσεις απόρριψης είναι συχνότερα βέλτιστες, κάτι που είναι αναμενόμενο, καθώς ένα μικρό κόστος εκκρεμών παραγγελιών δίνει την ευκαιρία να παρατείνουμε τους χρόνους παράδοσης των παραγγελιών με την απόρριψη αντικειμένων σχετικά χαμηλής ποιότητας, ενώ το υψηλότερο κόστος εκκρεμών παραγγελιών τείνει να κάνει πιο επείγουσα την όσο το δυνατόν ταχύτερη εκτέλεση των παραγγελιών με την ποιότητα να γίνεται δευτερεύον μέλημα. Η βέλτιστη πολιτική είναι επίσης ευαίσθητη στις μεταβολές του λ_1. Όσο μικρότερος είναι ο ρυθμός άφιξης των τακτικών πελατών (και παρόλα αυτά υψηλότερος από το λ_2), τόσο μικρότερο είναι το πλήθος εκκρεμών παραγγελιών και τόσο ευκολότερο είναι για το σύστημα να απορρίπτει προϊόντα, ώστε να έχει όσο το δυνατόν περισσότερους ικανοποιημένους (και τακτικούς) πελάτες, αποφεύγοντας παράλληλα τις υπερβολικές καθυστερήσεις στην εκτέλεση των παραγγελιών των πελατών.



Σχήμα 2.4 Βέλτιστες πολιτικές για $q = 4$ σε σχέση με το μοναδιαίο κόστος εκκρεμών παραγγελιών b_3 και τον μέσο ρυθμό άφιξης τακτικών πελατών λ_1.

Εξετάσαμε επίσης αριθμητικά τις επιπτώσεις της μεταβολής του ρυθμού παραγωγής μ_3 και του ρυθμού άφιξης παραγγελιών λ_2 στη βέλτιστη πολιτική. Σε όλες τις περιπτώσεις, υπήρχε μόνο ένα επίπεδο ποιότητας στο οποίο και οι δύο αποφάσεις είναι βέλτιστες, διαχωρισμένες από μια καμπύλη εναλλαγής αποφάσεων. Ωστόσο, υπάρχουν κάποιες περιπτώσεις στις οποίες και οι δύο αποφάσεις είναι παρούσες σε πολλά επίπεδα ποιότητας, όπως θα δούμε στη συνέχεια.

2.3 Μια ευρετική πολιτική τύπου κατωφλίου

Η αριθμητική διερεύνηση της προηγούμενης ενότητας δείχνει ότι η βέλτιστη πολιτική ελέγχου ποιότητας δεν έχει απλή δομή, διότι, εκτός από το επίπεδο ποιότητας των προϊόντων, εξαρτάται επίσης από το διάνυσμα κατάστασης n. Αυτό κάνει την βέλτιστη πολιτική δύσχρηστη, καθώς οι ανθρώπινοι χειριστές προτιμούν να χρησιμοποιούν απλούς κανόνες από περίπλοκες λύσεις που παράγονται από υπολογιστές. Παρ'όλα αυτά, υπάρχουν ορισμένες τιμές του q για τις οποίες η βέλτιστη πολιτική είναι ανεξάρτητη από το n. Στο προηγούμενο παράδειγμα, πάντα πωλούμε αντικείμενα εξόδου με επίπεδα ποιότητας $q = 0, 1, 2, 3$ και πάντα απορρίπτουμε αντικείμενα με $q = 5, 6, 7$. Η μόνη εξαίρεση είναι όταν $q = 4$, όπου και οι δύο αποφάσεις είναι εφαρμόσιμες και η βέλτιστη πολιτική έχει τη μορφή μιας καμπύλης εναλλαγής αποφάσεων όπως φαίνεται στα Σχήματα 2.2–2.4.

Σε αυτή την ενότητα, εξετάζουμε μια ευρετική πολιτική τύπου κατωφλίου για ευκολία στην υλοποίηση, ως εναλλακτική της βέλτιστης πολιτικής. Υποθέτουμε ότι το Q είναι ένα κατώφλι ποιότητας ($Q \leq L$), έτσι ώστε όλα τα αντικείμενα με $q > Q$ να απορρίπτονται, διαφορετικά αποστέλλονται στους αναμένοντες πελάτες. Στη συνέχεια, το πρόβλημα είναι η εύρεση του Q που μεγιστοποιεί τον μέσο ρυθμό κέρδους. Το κύριο πλεονέκτημα της προτεινόμενης πολιτικής είναι η υπολογιστική της αποδοτικότητα. Για περίπλοκα δίκτυα παραγωγής, η βέλτιστη πολιτική δεν μπορεί να υπολογιστεί λόγω της έκρηξης του χώρου καταστάσεων. Η εύρεση της βέλτιστης πολιτικής είναι απαιτητική υπολογιστικά ακόμα και για μια γραμμή παραγωγής δύο σταδίων. Από την άλλη πλευρά, η ευρετική πολιτική απαιτεί λιγότερους υπολογισμούς και μπορεί να εφαρμοστεί σε συστήματα παραγωγής με αυξημένη πολυπλοκότητα. Εκτός από το υπολογιστικό πλεονέκτημα, η προτεινόμενη πολιτική ορίου είναι εφαρμόσιμη ακόμη και όταν η εταιρεία παραγωγής έχει μόνο μερική πληροφόρηση για την κατάσταση της αγοράς, π.χ., όταν είναι γνωστό το M αλλά τα μερίδια αγοράς n_1 και n_2 που αντιστοιχούν σε τακτικούς και περιστασιακούς πελάτες είναι άγνωστα.

2.3.1 Εκτίμηση μέτρων απόδοσης

Όταν το σύστημα λειτουργεί υπό την προτεινόμενη πολιτική ελέγχου ποιότητας, μπορεί να μοντελοποιηθεί ως κλειστό δίκτυο ουρών αναμονής (ΚΔΑ) με N κόμβους και M εργασίες. Κάθε κόμβος i, $i = 3, ..., N$, του ΚΔΑ έχει ένα μόνο εκθετικό εξυπηρετητή με μέσο ρυθμό εξυπηρέτησης μ_i ίσο με τον μέσο ρυθμό παραγωγής της μηχανής i της μονάδας παραγωγής. Οι κόμβοι 1 και 2 αντιπροσωπεύουν αντίστοιχα τους μηχανισμούς που παράγουν αφίξεις τακτικών και περιστασιακών πελατών και σε κάθε χρονική στιγμή έχουν τόσους ενεργούς εξυπηρετητές όσες είναι οι εργασίες στις ουρές τους. Οι διακομιστές σε αυτούς τους δύο κόμβους έχουν εκθετικά κατανεμημένους χρόνους επεξεργασίας με μέσες τιμές $1/\lambda_i$, για $i = 1, 2$. Το ΚΔΑ και το σύστημα παραγωγής έχουν την ίδια σχηματική αναπαράσταση όπως φαίνεται στο Σχήμα 2.1.

Σε οποιαδήποτε χρονική στιγμή, το διάνυσμα κατάστασης n του συστήματος παραγωγής είναι ίδιο με το διάνυσμα του πλήθους αριθμό εργασιών στους κόμβους του ΚΔΑ. Οι πιθανότητες δρομολόγησης του κόμβου N εξαρτώνται από την ποιότητα του αντίστοιχου εξερχόμενου προϊόντος και την απόφαση που λαμβάνεται στο σύστημα παραγωγής. Υπό την προτεινόμενη πολιτική, η ποιότητα του τελικού προϊόντος q προσδιορίζεται με επιθεώρηση και αν $q > Q$ (χαμηλή ποιότητα), τότε αυτό το αντικείμενο απορρίπτεται και μια νέα πρώτη ύλη απελευθερώνεται στην πρώτη μηχανή της μονάδας παραγωγής. Στην ουσία, αυτό είναι το ίδιο με μια εργασία που δρομολογείται από τον κόμβο N πίσω στον κόμβο 3, όπως φαίνεται στο Σχήμα 2.1. Η αντίστοιχη πιθανότητα δρομολόγησης δίδεται από την εξίσωση $p_{N,3} = p_{Q+1} + \ldots + p_L$. Επιπλέον, αν το τελικό προϊόν έχει αποδεκτό επίπεδο ποιότητας ($q \leq Q$), τότε θα αγοραστεί από έναν πελάτη. Με πιθανότητα $p_{N,1} = p_0 s_0 + \ldots + p_Q s_Q$ ο πελάτης που αγοράζει το προϊόν θα είναι ικανοποιημένος. Στο ισοδύναμο ΚΔΑ, μια εργασία αναχωρεί από τον κόμβο N και δρομολογείται στον κόμβο 1 (ο ικανοποιημένος πελάτης γίνεται τακτικός πελάτης). Ωστόσο, με πιθανότητα $p_{N,2} = p_0(1 - s_0) + \ldots + p_Q(1 - s_Q)$ ο πελάτης θα είναι δυσαρεστημένος και στο ισοδύναμο ΚΔΑ η εργασία θα πάει στον κόμβο 2 (ο δυσαρεστημένος πελάτης γίνεται περιστασιακός πελάτης). Οι άλλες πιθανότητες δρομολόγησης του ΚΔΑ είναι ίδιες με αυτές της μονάδας παραγωγής. Τέλος, όλες οι εργασίες που προέρχονται από τους κόμβους 1 και 2 δρομολογούνται στον κόμβο 3 του ΚΔΑ, δηλαδή, $p_{1,3} = p_{2,3} = 1$.

Στη συνέχεια συνοψίζουμε έναν γνωστό αλγόριθμο από τη θεωρία ουρών, ο οποίος επιτρέπει τον υπολογισμό των πιθανοτήτων μόνιμης κατάστασης $P(n) = P(n_1, \ldots, n_N)$ και όλων των συνιστωσών του μέσου ρυθμού κέρδους του συστήματος.

Ας ονομάσουμε TH_N το μέσο ρυθμό παραγωγής του κόμβου N, $B_i = E(n_i)$ το μέσο απόθεμα/έλλειμμα του κόμβου i, $\Pi = [p_{i,j}]$ τον πίνακα πιθανοτήτων δρομολόγησης και $u = [u_1 \ldots u_N]$ οποιαδήποτε μη αρνητική λύση του συστήματος των γραμμικών εξισώσεων $u = u\Pi$. Το διάνυσμα u καθορίζεται ως γινόμενο μιας σταθεράς, έτσι είμαστε ελεύθεροι να επιλέξουμε ένα σχήμα κανονικοποίησης για τα u_i (π.χ. $u_N = 1$ ή $u_1 + \ldots + u_N = 1$). Ας υποθέσουμε ότι $\alpha_i(n_i)$ είναι ο αριθμός των κατειλημμένων εξυπηρετητών στον κόμβο i. Έχουμε ότι $\alpha_i(n_i) = 1$ για όλους τους κόμβους $i \geq 3$ και $n_i \geq 1$, δεδομένου ότι ο κάθε ένας από αυτούς τους κόμβους έχει μόνο έναν εξυπηρετητή, ενώ για τους κόμβους 1 και 2 έχουμε ότι $\alpha_1(n_1) = n_1$ και $\alpha_2(n_2) = n_2$. Στη συνέχεια, υπολογίζουμε αναδρομικά τις ακολουθίες $\beta_i(n_i) = \alpha_i(n_i)\beta_i(n_i - 1)$ χρησιμοποιώντας την $\beta_i(0) = 1$ ως οριακή τιμή. Αποδεικνύεται ότι $\beta_i(n_i) = 1$ για $i \geq 3$ και $\beta_i(n_i) = n_i!$ για $i = 1, 2$. Οι πιθανότητες μόνιμης κατάστασης του συστήματος δίνονται από (βλ. π.χ. [6]).

$$P(n) = \frac{1}{G(M)} \prod_{i=1}^{N} \frac{\rho^{n_i}}{\beta_i(n_i)} \tag{2.3}$$

όπου $\rho_1 = u_1/\lambda_1$, $\rho_2 = u_2/\lambda_2$, $\rho_i = u_i/\mu_i$, και $G(M)$ είναι μια σταθερά κανονικοποίησης που προκύπτει από

$$G(M) = \sum_{n_1+\cdots+n_N=M} \prod_{i=1}^{N} \frac{\rho^{n_i}}{\beta_i(n_i)} \tag{2.4}$$

Τα μέτρα απόδοσης του συστήματος υπολογίζονται από γνωστούς τύπους των συστημάτων αναμονής ([6]). Ο μέσος ρυθμός παραγωγής του κόμβου $i \geq 3$ (δηλαδή μηχανής) είναι

$$\mathrm{TH}_i = \rho_i \frac{G(M-1)}{G(M)} \tag{2.5}$$

το μέσο απόθεμα και ο μέσος αριθμός εκκρεμών παραγγελιών που είναι στο στάδιο παραγωγής $i \geq 3$ είναι

$$B_i = \frac{1}{G(M)} \sum_{m=1}^{M} G(M-m) \rho_i^m$$

και ο συνολικός μέσος ρυθμός κέρδους του συστήματος

$$J = r(1 - p_{N,3}) \mathrm{TH}_N - c p_{N,3} \mathrm{TH}_N - \sum_{i=3}^{N} b_i B_i$$

Η τιμή J εξαρτάται από την επιλογή του επιπέδου ποιότητας $Q = 0, \ldots, L$, που καθορίζει τις πιθανότητες δρομολόγησης p_{3i}. Για να βρούμε το βέλτιστο Q πραγματοποιούμε εξαντλητική αναζήτηση.

1) Αρχικοποιούμε την βέλτιστη πολιτική, ορίζοντας $J^* = -\infty$ και το όριο ποιότητας $Q = 0$.

2) Για $Q = 1, \ldots, L$:

α) Υπολογίζουμε τις αντίστοιχες πιθανότητες δρομολόγησης $p_{N,1}$, $p_{N,2}$, και $p_{N,3}$, καθώς και τις πιθανότητες μόνιμης κατάστασης $P(n)$ του ΚΔΑ και τον συνολικό μέσο ρυθμό κέρδους J του συστήματος.

β) Αν $J^* > J$, η τρέχουσα Q είναι η τοπικά βέλτιστη απόφαση: αποθηκεύουμε ορίζουμε $J^* = J$ και $Q^* = Q$.

3) Επαναλαμβάνουμε τα παραπάνω βήματα μέχρι να εξετασθούν όλα τα επίπεδα ποιότητας $Q \leq L$. Το βέλτιστο ζεύγος είναι (Q^*, J^*).

2.4 Αριθμητικά αποτελέσματα

Σε αυτή την ενότητα, συγκρίνουμε αριθμητικά την προτεινόμενη πολιτική με τη βέλτιστη πολιτική, για να δούμε αν η πρώτη μπορεί να χρησιμοποιηθεί ως μια εύκολα υλοποιήσιμη εναλλακτική της δεύτερης με μικρές αποκλίσεις στην απόδοση.

2.4.1 Εφαρμογή σε συστήματα ενός σταδίου

Πρώτα εξετάζουμε ένα σύστημα παραγωγής ενός σταδίου, όπως αυτό που περιγράφεται στην Παράγραφο 2.2. Υποθέτουμε ότι $M = 50$ και τις ίδιες πιθανότητες επιπέδου ποιότητας p_q και πιθανότητες ικανοποίησης s_q όπως στην Παράγραφο 2.2 (Πίνακας 2.1). Ο Πίνακας 2.2 δείχνει τις τιμές των υπόλοιπων παραμέτρων και τις αντίστοιχες βέλτιστες πολιτικές. Η στήλη «Δομή πολιτικής» περιγράφει την βέλτιστη πολιτική για κάθε περίπτωση προβλήματος. Με λίγες εξαιρέσεις, για επίπεδα ποιότητας κοντά στο 0 και κοντά στο 7, η βέλτιστη απόφαση είναι, αντίστοιχα, να δεχτεί (1) και να απορρίψει (0) το αντικείμενο ανεξάρτητα από την κατάσταση του συστήματος n.

Τα επίπεδα ποιότητας για τα οποία η βέλτιστη απόφαση εξαρτάται από το n υποδεικνύονται στον Πίνακα 2.2 με το γράμμα M. Για παράδειγμα, η καταχώρηση 1111M000 στην πρώτη γραμμή του πίνακα υποδεικνύει ότι η απόφαση πώλησης (1) είναι βέλτιστη για $q = 0, \ldots, 3$ και η απόφαση απόρριψης (0) είναι βέλτιστη για $q = 5, 6, 7$ ανεξαρτήτως του n, ενώ για $q = 4$ έχουμε δύο χωριστές περιοχές του διανυσματικού χώρου όπου κάθε απόφαση είναι βέλτιστη. Παρατηρούμε ότι το όριο Q της βέλτιστης ευρετικής πολιτικής ισούται πάντα με ένα λιγότερο από τον αριθμό των 1 της βέλτιστης απόφασης. Θυμηθείτε ότι το πρώτο bit 1 αντιστοιχεί στο $q = 0$. Αυτό σημαίνει ότι το Q συμπίπτει με το τελευταίο επίπεδο ποιότητας για το οποίο η απόφαση πώλησης είναι πάντα βέλτιστη. Επιπλέον, βλέπουμε ότι οι μέσοι ρυθμοί κέρδους των δύο πολιτικών είναι ίσοι έως δύο δεκαδικά ψηφία. Έτσι, για τα σύνολα δεδομένων που χρησιμοποιήθηκαν σε αυτή τη μελέτη, η προτεινόμενη πολιτική φαίνεται να είναι μια πολύ καλή προσέγγιση της βέλτιστης πολιτικής. Παρατηρούμε επίσης ότι τα αποδεκτά όρια ποιότητας και στις δύο πολιτικές είναι μάλλον ανεξάρτητα από τις παραλλαγές των παραμέτρων, εκτός από τη λ_2. Καθώς η λ_2 αυξάνεται, οι περιστασιακοί πελάτες τείνουν να έχουν παρόμοια αγοραστική συμπεριφορά με τους τακτικούς πελάτες και, συνεπώς, ένα υψηλό επίπεδο ρυθμού πωλήσεων μπορεί να διατηρηθεί ακόμα και με μεγάλο μερίδιο περιστασιακών πελατών, πωλώντας προϊόντα χαμηλότερης ποιότητας και αποφεύγοντας τα έξοδα απόρριψης.

Πίνακας 2.2 Σύγκριση πολιτικών για σύστημα ενός σταδίου και διάφορες τιμές των παραμέτρων του

Τιμές παραμέτρων						**Βέλτιστη πολιτική**		**Πολιτική κατωφλίου**	
λ_1	λ_2	μ	c	r	b	**Δομή πολιτικής**	**Μέσο κέρδος**	**Μέσο κέρδος**	**Κατώφλι ποιότητας**
1	0.1	60	3.5	4	0.3	1111M000	77.44	77.44	επίπεδο 3
1	0.1	60	3.5	4	0.4	1111M000	77.15	77.15	επίπεδο 3
1	0.1	60	3.5	4	0.9	1111M000	76.87	76.87	επίπεδο 3
1	0.1	60	3.5	4	1.2	1111MM00	76.59	76.59	επίπεδο 3
1	0.1	60	3.5	4	0.3	1111M000	77.44	77.44	επίπεδο 3
1	0.1	60	3.5	3	0.3	11111M00	53.92	53.92	επίπεδο 4

Τιμές παραμέτρων						Βέλτιστη πολιτική		Πολιτική κατωφλίου	
λ_1	λ_2	μ	c	r	b	**Δομή πολιτικής**	**Μέσο κέρδος**	**Μέσο κέρδος**	**Κατώφλι ποιότητας**
1	0.1	60	3.5	4.8	0.3	`1111M000`	96.75	96.75	επίπεδο 3
1	0.1	60	3.5	4.5	0.3	`1111M000`	89.51	89.51	επίπεδο 3
1	0.1	60	3.5	4	0.3	`1111M000`	77.44	77.44	επίπεδο 3
1	0.1	60	3	4	0.3	`1111M000`	80.13	80.13	επίπεδο 3
1	0.1	60	4	4	0.3	`1111M000`	74.74	74.74	επίπεδο 3
1	0.1	60	2.5	4	0.3	`11110000`	82.82	82.82	επίπεδο 3
1	0.1	60	3.5	4	0.3	`1111M000`	77.44	77.44	επίπεδο 3
0.9	0.1	60	3.5	4	0.3	`1111M000`	73.98	73.98	επίπεδο 3
1.1	0.1	60	3.5	4	0.3	`1111M000`	80.50	80.50	επίπεδο 3
0.8	0.1	60	3.5	4	0.3	`1111M000`	70.07	70.07	επίπεδο 3
1	0.1	60	3.5	4	0.3	`1111M000`	77.44	77.44	επίπεδο 3
1	0.08	60	3.5	4	0.3	`1111M000`	68.15	68.15	επίπεδο 3
1	0.15	60	3.5	4	0.3	`1111MMM0`	94.59	94.59	επίπεδο 3
1	0.2	60	3.5	4	0.3	`11111MMM`	108.83	108.83	επίπεδο 4
1	0.1	60	3.5	4	0.3	`1111M000`	77.44	77.44	επίπεδο 3
1	0.1	55	3.5	4	0.3	`1111M000`	77.12	77.12	επίπεδο 3
1	0.1	65	3.5	4	0.3	`1111M000`	77.67	77.67	επίπεδο 3
1	0.1	50	3.5	4	0.3	`1111M000`	76.69	76.69	επίπεδο 3

2.4.2 Γραμμή παραγωγής 2 σταδίων

Δοκιμάζουμε τώρα την πολιτική κατωφλίου σε μια γραμμή παραγωγής δύο μηχανών. Όπως και προηγουμένως, χρησιμοποιούμε οκτώ επίπεδα ποιότητας και τις ίδιες πιθανότητες p_q και s_q (Πίνακας 2.1). Για να αποφύγουμε το πρόβλημα της έκρηξης του χώρου καταστάσεων, υποθέτουμε ότι η αγορά αποτελείται από συνολικά $M = 5$ πελάτες.

Ο Πίνακας 2.3 δείχνει τις υπόλοιπες τιμές των παραμέτρων του συστήματος και τις αντίστοιχες βέλτιστες παραμέτρους των πολιτικών και τις τιμές του μέσου ρυθμού κέρδους.

Πίνακας 2.3 Σύγκριση πολιτικών για σύστημα δύο σταδίων και διάφορες τιμές των παραμέτρων του

Τιμές παραμέτρων								**Βέλτιστη πολιτική**		**Πολιτική κατωφλίου**	
λ_1	λ_2	μ_3	μ_4	c	r	b_3	b_4	**Δομή πολιτικής**	**Μέσο κέρδος**	**Μέσο κέρδος**	**Κατώφλι ποιότητας**
1	0.1	6	5.5	3.5	4	0.25	0.55	11111M00	5.94	5.94	level 4
1	0.1	6	5.5	2.5	4	0.25	0.55	1111M000	**6.23**	**6.22**	level 3
1	0.1	6	5.5	3	4	0.25	0.55	1111M000	**6.05**	**6.03**	level 4
1	0.1	6	5.5	4	4	0.25	0.55	11111MMM	**5.85**	**5.84**	level 4
1	0.1	6	5.5	3.5	3	0.25	0.55	11111MMM	**4.24**	**4.23**	level 5
1	0.1	6	5.5	3.5	3.5	0.25	0.55	11111MMM	5.07	5.07	level 4
1	0.1	6	5.5	3.5	5	0.25	0.55	1111M000	**7.75**	**7.73**	level 3
1	0.1	6	5.5	3.5	4	0.2	0.5	11111M00	5.98	5.98	level 4
1	0.1	6	5.5	3.5	4	0.3	0.6	11111M00	5.89	5.89	level 4
1	0.1	6	5.5	3.5	4	0.35	0.65	11111M00	5.85	5.85	level 4
1	0.1	6	5.5	3.5	4	0.25	0.45	11111M00	5.98	5.98	level 4
1	0.1	6	5.5	3.5	4	0.25	0.5	11111M00	5.96	5.96	level 4
1	0.1	6	5.5	3.5	4	0.25	0.6	11111M00	5.91	5.91	level 4
0.8	0.1	6	5.5	3.5	4	0.25	0.55	11111MM0	5.56	5.56	level 4
0.9	0.1	6	5.5	3.5	4	0.25	0.55	11111M00	5.76	5.76	level 4
1.1	0.1	6	5.5	3.5	4	0.25	0.55	11111M00	6.09	6.09	level 4
1	0.08	6	5.5	3.5	4	0.25	0.55	1111M000	**5.29**	**5.27**	level 4
1	0.15	6	5.5	3.5	4	0.25	0.55	11111MMM	**7.17**	**7.16**	level 5
1	0.2	6	5.5	3.5	4	0.25	0.55	11111111	8.14	8.14	level 7
1	0.1	5	5.5	3.5	4	0.25	0.55	11111M00	5.81	5.81	level 4
1	0.1	5.5	5.5	3.5	4	0.25	0.55	11111M00	5.88	5.88	level 4
1	0.1	6.5	5.5	3.5	4	0.25	0.55	11111M00	5.98	5.98	level 4
1	0.1	6	4.5	3.5	4	0.25	0.55	11111MMM	5.70	5.70	level 4
1	0.1	6	5	3.5	4	0.25	0.55	11111MMM	5.83	5.83	level 4
1	0.1	6	6.5	3.5	4	0.25	0.55	1111M000	6.09	6.09	level 4

Βλέπουμε ότι το όριο της βέλτιστης ευρετικής πολιτικής, στις περισσότερες περιπτώσεις, συμπίπτει με το τελευταίο επίπεδο ποιότητας που επιστρέφει 1 ως βέλτιστη απόφαση για κάθε κατάσταση του συστήματος στη βέλτιστη πολιτική. Υπάρχουν περιπτώσεις στις οποίες αυτό δεν ισχύει. Ο λόγος για αυτό μπορεί να είναι το σχετικά μικρό μέγεθος του πληθυσμού M ή ο μεγάλος αριθμός επιπέδων ποιότητας που καθιστά τη διάκριση μεταξύ τους λιγότερο σαφή. Όπως και προηγουμένως, σε σύγκριση με τη βέλτιστη πολιτική που λαμβάνεται με την μέθοδο των διαδοχικών προσεγγίσεων, η προτεινόμενη πολιτική κατωφλίου επιτυγχάνει σχεδόν τα ίδια επίπεδα μέσου κέρδους με ελάχιστο υπολογιστικό κόστος. Μικρές διαφορές παρατηρούνται σε μερικές περιπτώσεις που επισημαίνονται με έντονη γραφή στον πίνακα. Όπως και στην περίπτωση ενός σταδίου, όταν το λ_2 αυξάνεται, είναι βέλτιστο για το σύστημα να πωλεί προϊόντα χαμηλότερης ποιότητας, αποφεύγοντας έτσι ένα μέρος των εξόδων απόρριψης και αυξάνοντας τον πληθυσμό των περιστασιακών πελατών χωρίς να θέτει σε κίνδυνο τις μελλοντικές πωλήσεις.

2.5 Σύνοψη

Σε αυτό το κεφάλαιο χρησιμοποιήσαμε εργαλεία από τον χώρο του σταχαστικού δυναμικού προγραμματισμού και την θεωρία ουρών αναμονής για να μοντελοποιήσουμε την επίδραση των διακυμάνσεων της ποιότητας των προϊόντων στην ικανοποίηση των πελατών, στο μερίδιο αγοράς και στην κερδοφορία σε μια κατηγορία συστημάτων παραγωγής κατά παραγγελία. Είδαμε ότι τα αποδεκτά επίπεδα ποιότητας που μεγιστοποιούν την κερδοφορία δεν είναι σταθερά, αλλά εξαρτώνται από την κατάσταση στη μονάδα παραγωγής καθώς και από το μερίδιο αγοράς (τακτικοί πελάτες) της εταιρείας. Για καταστάσεις στις οποίες είναι επιθυμητή η διατήρηση ενός σταθερού επιπέδου ποιότητας για τα προϊόντα ή όταν υπάρχει μερική πληροφορία για την αγορά, προτείνεται μια απλή πολιτική ελέγχου τύπου κατωφλίου, η οποία έχει λογικές υπολογιστικές απαιτήσεις και φαίνεται να είναι μια πολύ καλή προσέγγιση της βέλτιστης πολιτικής.

Στο πρόβλημα που μελετήσαμε εδώ η ικανοποίηση των πελατών είναι ανεξάρτητη από την κατηγορία, στην οποία ανήκαν πριν αγοράσουν προϊόν. Στο επόμενο κεφάλαιο μελετάμε συστήματα, όπου οι πελάτες έχουν μνήμης της κατηγορίας στην οποία ανήκαν πριν εισέλθουν στο σύστημα παραγωγής και διερευνούμε πως αυτό επηρεάζει την μελλοντική τους ικανοποίηση.

Ιδιαίτερο ενδιαφέρον παρουσιάζει η απόδοση χρόνου παράδοσης παραγγελιών (καθυστερήσεις πελατών) στη μοντελοποίηση της ικανοποίησης των πελατών και η διερεύνιση της αλληλεπίδραση με τις πολιτικές παραγωγής και ελέγχου ποιότητας. Τέτοιες επεκτάσεις θέτουν τόσο υπολογιστικές όσο και θεωρητικές προκλήσεις λόγω του μεγάλου χώρου καταστάσεων και της παραβίασης της ιδιότητας της μνήμης, που επιτρέπει αποδοτικούς αλγόριθμους λύσης προβλημάτων ουρών. Η δυσκολία μελέτης τέτοιων συστημάτων οδηγεί στην σκέψη χρησιμοποίησης προσεγγιστικών τεχνικών όπως οι μεθοδολογίες ενισχυτικής μάθησης. Άλλες πιο άμεσες επεκτάσεις του μοντέλου θα ήταν η ένταξη περισσότερων από δύο καταστάσεων ικανοποίησης πελατών, η ανάλυση διαφορετικών συμπεριφορών πελατών και εναλλακτικών λειτουργιών για τη σύνδεση των p_q και s_q με το q, και η εξέταση του ανταγωνισμού στην αγορά και η μελέτη διαφορετικών εντάσεων ανταγωνισμού (π.χ., διαφορετικά πρότυπα ικανοποίησης πελατών, διαφορετικοί συνδυασμοί μέσων ρυθμών ζήτησης). Τέλος, πιο σύνθετα συστήματα με τελικούς και ενδιάμεσους σταθμούς επιθεώρησης, που υπόκεινται σε σφάλματα επιθεώρησης και τοπικά επισκευάσιμα ελαττωματικά προϊόντα εκτός από την απόρριψη αντικειμένων, μπορούν να μοντελοποιηθούν με την επέκταση των μοντέλων που παρουσιάζονται εδώ και στο [11]

3 Σχέση ποιότητας προϊόντων και κερδοφορίας σε απλά συστήματα παραγωγής κατά παραγγελίες όταν η συμπεριφορά καταναλωτών παρουσιάζει μνήμη

3.1 Εισαγωγή

Στο προηγούμενο κεφάλαιο μελετήσαμε συστήματα παραγωγής με πεπερασμένη αγορά που αποτελείται από τακτικούς πελάτες με υψηλά ποσοστά ζήτησης και περιστασιακούς πελάτες με χαμηλότερη ζήτηση, υποθέτοντας ότι η ικανοποίηση είναι συνάρτηση της ποιότητας του προϊόντος ανεξάρτητα από την κατηγορία του πελάτη. Στόχος μας εδώ είναι να μελετήσουμε την επίδραση της ικανοποίησης στην κερδοφορία, όταν εξαρτάται τόσο από την κατηγορία στην οποία ανήκουν οι πελάτες όσο και από την ποιότητα των παραγόμενων προϊόντων. Εξετάζουμε ένα Μαρκοβιανό σύστημα παραγωγής κατά παραγγελίες (Make-to Order) ενός σταδίου, που παράγει για να καλύψει παραγγελίες από τακτικούς και περιστασιακούς πελάτες. Ένας πελάτης που αγοράζει ένα προϊόν μπορεί να γίνει τακτικός ή περιστασιακός πελάτης, με πιθανότητες που εξαρτώνται τόσο από το επίπεδο ποιότητας του προϊόντος όσο και από την προηγούμενη κατάσταση του πελάτη. Χρησιμοποιούμε στοχαστικό δυναμικό προγραμματισμό για να διερευνήσουμε αριθμητικά τη δομή της κοινής πολιτικής παραγωγής και ελέγχου ποιότητας που μεγιστοποιεί τον μέσο ρυθμό κέρδους του συστήματος. Εξετάζουμε επίσης πώς η γνώση της προηγούμενης κατάστασης του πελάτη επηρεάζει τη βέλτιστη πολιτική. Η αριθμητική ανάλυση παρέχει χρήσιμες πληροφορίες και υποδείξεις για την εκπόνηση μιας απλής αλλά αποτελεσματικής ευρετικής πολιτικής.

Το κεφάλαιο είναι διαρθρωμένο ως εξής. Στην Παράγραφο 3.2 περιγράφουμε ένα σύστημα παραγωγής ενός σταδίου που παράγει κατά παραγγελία με συμπεριφορές πελατών εξαρτώμενες από το επίπεδο ποιότητας των τελικών προϊόντων αλλά και από την προηγούμενη κατηγορία στην οποία ανήκαν πριν εισέλθουν στο σύστημα. Το πρόβλημα περιγράφεται ως ένα πρόβλημα στοχαστικού δυναμικού προγραμματισμού. Μια αριθμητική διερεύνηση της βέλτιστης πολιτικής παρουσιάζεται στην Παράγραφο 3.3. Τα αποτελέσματα των αριθμητικών πειραμάτων δείχνουν ότι η βέλτιστη πολιτική απαιτεί ακριβή γνώση της τρέχουσας κατάστασης της αγοράς, δηλαδή του πλήθους των τακτικών και περιστασιακών πελατών αλλά και του πλήθους εκκρεμών παραγγελιών κάθε κατηγορίας εντός του συστήματος. Όπως και στο προηγούμενο κεφάλαιο προτείνουμε μια απλή ευρετική πολιτική τύπου κατωφλίου που παρουσιάζεται στην Παράγραφο 3.4. Στην ίδια παράγραφο αναλύουμε πως η πολιτική αυτή μπορεί να περιγραφεί μαθηματικά με την χρήση κλειστών δικτύων αναμομής που εξυπηρετούν πολλές κατηγορίες πελατών. Στην Παράγραφο 3.5 γίνεται μια αριθμητική σύγκριση ανάμεσα στην βέλτιστη και στην προτεινόμενη ευρετική πολιτική. Τέλος στην Παράγραφο 3.6 παρουσιάζονται τα συμπεράσματα και πιθανές επεκτάσεις.

3.2 Περιγραφή συστήματος, μοντελοποίηση και λήψη βέλτιστων αποφάσεων

Εξετάζουμε μια μονάδα παραγωγής ενός σταδίου, που παράγει ένα μόνο προϊόν, η οποία λειτουργεί μόνο όταν υπάρχουν πελάτες που περιμένουν για εξυπηρέτηση (σύστημα παραγωγής κατά παραγγελίες). Το μέγεθος της αγοράς M θεωρείται γνωστό και σταθερό. Οι M πελάτες χωρίζονται σε δύο διακριτές κατηγορίες: την κατηγορία των τακτικών πελατών, με συμβολισμό $k = 1$, και την κατηγορία των περιστασιακών πελατών, $k = 2$. Η κατάσταση του συστήματος μπορεί να περιγραφεί από τρεις μεταβλητές:

x πλήθος τακτικών πελατών που περιμένουν εξυπηρέτηση εντός του συστήματος,

y πλήθος περιστασιακών πελατών που περιμένουν εξυπηρέτηση εντός του συστήματος,

z πλήθος τακτικών πελατών εκτός συστήματος.

Εφόσον ο συνολικός πληθυσμός πελατών είναι M προκύπτει ότι το πλήθος περιστασιακών πελατών εκτός συστήματος παραγωγής είναι $M - x - y - z$.

Οι αφίξεις πελατών είναι τυχαίες και ακολουθούν την κατανομή Poisson, κάθε πελάτης ζητά μία μονάδα προϊόντος και έχει ρυθμό ζήτησης που εξαρτάται από την κατηγορία του, λ_k, $k = 1, 2$, όπου $\lambda_1 > \lambda_2$. Οι χρόνοι επεξεργασίας των προϊόντων είναι ανεξάρτητες τυχαίες μεταβλητές εκθετικά κατανεμημένες με μέση τιμή $1/\mu$. Κάθε εξερχόμενο προϊόν έχει ένα επίπεδο ποιότητας i ($i = 1, \ldots, I$ με την ποιότητα να μειώνεται καθώς αυξάνεται το i) με αντίστοιχη πιθανότητα p_i. Κάθε φορά που παράγεται ένα προϊόν αποφασίζουμε αν θα το πουλήσουμε σε έναν τακτικό πελάτη που περιμένει, ή σε έναν περιστασιακό πελάτη που περιμένει, ή αν θα το απορρίψουμε. Αν ένα προϊόν ποιότητας επιπέδου i πωληθεί σε πελάτη κατηγορίας k, τότε αυτός ο πελάτης θα είναι τελικά ικανοποιημένος με πιθανότητα s_i^k ή δυσαρεστημένος με την συμπληρωματική πιθανότητα. Οι ικανοποιημένοι πελάτες εντάσσονται στην κατηγορία των τακτικών πελατών, ενώ οι δυσαρεστημένοι γίνονται περιστασιακοί πελάτες. Υποθέτουμε ότι τα προϊόντα υψηλής ποιότητας (i = 1) αντιστοιχούν σε υψηλότερες πιθανότητες ικανοποίησης.

Το συνολικό μέτρο απόδοσης του συστήματος είναι ο μέσος ρυθμός κέρδους. Αυτή η ποσότητα εξαρτάται από το ρυθμό πωλήσεων (που ισούται με τον ρυθμό παραγωγής του συστήματος), το κόστος των απορριπτόμενων προϊόντων και το κόστος αναμονής. Εξετάζουμε τέσσερις οικονομικές παραμέτρους:

- p κέρδος από την πώληση μιας μονάδας προϊόντος,
- r μοναδιαίο κόστος απόρριψης (κόστος από την απόρριψη ή/και επανακατεργασία ενός προϊόντος),
- b_i μοναδιαίο κόστος αναμονής ανά πελάτη κατηγορίας i και ανά μονάδα χρόνου, (i = 1, 2).

Η κατάσταση του συστήματος περιγράφεται από το σύνολο μεταβλητών (x, y, z) με χώρο καταστάσεων $S = \{(x, y, z) \mid x = 0, 1, \ldots, M, y = 0, \ldots, M - x$ and $z = 0, \ldots, M - x - y\}$. Δεδομένου ότι ο αριθμός των πελατών στο σύστημα είναι πεπερασμένος και ίσος με M, ο αριθμός των περιστασιακών πελατών εκτός συστήματος παραγωγής θα είναι $M - x - y - z$. Σε κάθε χρονική στιγμή ολοκλήρωσης παραγωγής λαμβάνουμε μια απόφαση πώλησης $\pi(x, y, z, i)$, όπου $\pi(x, y, z, i) = 1$ όταν αποφασίζουμε να πουλήσουμε το προϊόν ποιότητας επιπέδου i σε τακτικό πελάτη που περιμένει, $\pi(x, y, z, i) = 2$ όταν αποφασίζουμε να πουλήσουμε το προϊόν ποιότητας επιπέδου i σε περιστασιακό πελάτη που περιμένει και $\pi(x, y, z, i) = 3$ όταν αποφασίζουμε να απορρίψουμε ένα προϊόν ποιότητας επιπέδου i. Όταν απορρίπτεται ένα τελικό προϊόν, η κατάσταση του συστήματος δεν αλλάζει. Εάν αποφασίσουμε να πουλήσουμε ένα προϊόν σε πελάτη και ο πελάτης είναι ικανοποιημένος με την ποιότητα του αντικειμένου, τότε γίνεται επιπλέον τακτικός πελάτης εκτός συστήματος παραγωγής και η κατάσταση του συστήματος αλλάζει σε $(x-1, y, z+1)$ ή $(x, y-1, z+1)$ αντίστοιχα. Σε περίπτωση δυσαρέσκειας, ο πελάτης εντάσσεται στους περιστασιακούς πελάτες σε εκτός συστήματος παραγωγής και η κατάσταση του συστήματος αλλάζει σε $(x-1, y, z)$ *ή* $(x, y-1, z)$ αντίστοιχα. Η εξέλιξη του συστήματος καθορίζεται από δύο τύπους γεγονότων: την τοποθέτηση παραγγελίας από έναν πελάτη τύπου k και την παραγωγή ενός προϊόντος ποιότητας i. Χρησιμοποιώντας την τεχνική ομοιομορφοποίησης (βλ. [30], [35], [3] [37]) διατυπώνουμε το πρόβλημα ως διαδικασία λήψης αποφάσεων Markov. Το σταθερό μέγεθος M της αγοράς περιορίζει τον χώρο καταστάσεων και διασφαλίζει τη σταθερότητα του συστήματος. Η συνεχής αλυσίδα Markov ομογενοποιείται χρησιμοποιώντας τον μέγιστο ρυθμό μετάβασης $v = M\lambda_1 + \mu$, που είναι ένας ομοιόμορφος ρυθμός εμφάνισης γεγονότων και επιτρέπει τη μετατροπή του συνεχούς χρόνου σε ακολουθία σταδίων λήψης αποφάσεων. Ακολουθώντας την τυπική προσέγγιση του στοχαστικού δυναμικού προγραμματισμού, ορίζουμε το $V_k(x, y, z)$ ως το συνολικό αναμενόμενο κέρδος κατά τα πρώτα k γεγονότα (ή στάδια λήψης αποφάσεων) όταν η αρχική κατάσταση είναι (x, y, z). Οι εξισώσεις Bellman για το βέλτιστο συνολικό αναμενόμενο κέρδος κατά τα πρώτα $k + 1$ γεγονότα είναι:

$$
\begin{aligned}
V_{k+1}(x,y,z) = \frac{1}{v}\Big\{ & z\lambda_1 V_k\left(x+1,y,z-1\right) + \left(M-x-y-z\right)\lambda_2 V_k\left(x,y+1,z\right) + \\
& + \sum_{i=1}^{I} \mu p_i \max\Big[I_{x>0}\left(s_i^1 V_k\left(x-1,y,z+1\right) + \left(1-s_i^1\right)V_k\left(x-1,y,z\right) + p\right), \\
& \qquad I_{y>0}\left(s_i^2 V_k\left(x,y-1,z+1\right) + \left(1-s_i^2\right)V_k(x,y-1,z) + p\right), \\
& \qquad I_{x+y>0}\left(V_k\left(x,y,z\right) - r\right)\Big] + \\
& + \Big[\left(M-z\right)\left(\lambda_1-\lambda_2\right) + \left(x+y\right)\lambda_2 + \left(1-I\left(x+y>0\right)\right)\mu\Big] V_k\left(x,y,z\right) - b_1 x - b_2 y \Big\}
\end{aligned}
\tag{3.1}
$$

για $k > 0$ και $(x, y, z) \in S$, όπου I_C είναι μια δείκτρια συνάρτηση που ισούται με 1 εάν η συνθήκη C είναι αληθής και μηδέν διαφορετικά, και $V_0(x, y, z) = 0$ για κάθε $(x, y, z) \in S$. Η βέλτιστη μέση μακροχρόνια συνάρτηση κέρδους μπορεί να προσεγγιστεί αριθμητικά από τον αλγόριθμο διαδοχικών προσεγγίσεων όπως αναφέρεται στην Πρόταση 2.1 του Κεφαλαίου 2 (βλ. επίσης Puterman [35]). Βασική προϋπόθεση για να ισχύει η Πρόταση 2.1 είναι ότι κάθε βέλτιστη αναμενόμενη στάσιμη ντετερμινιστική πολιτική έχει έναν απεριοδικό πίνακα μεταβάσεων. Μια κατάσταση λέμε ότι είναι απεριοδική αν οι επιστροφές σε αυτήν την κατάσταση μπορούν να συμβούν σε άτακτα χρονικά διαστήματα. Στο σύστημά μας, ο πίνακας μεταβάσεων κάθε πολιτικής είναι απεριοδικός επειδή η πιθανότητα να παραμείνει το σύστημα στην τρέχουσα κατάσταση είναι θετική ανεξάρτητα από την απόφαση που λαμβάνεται.

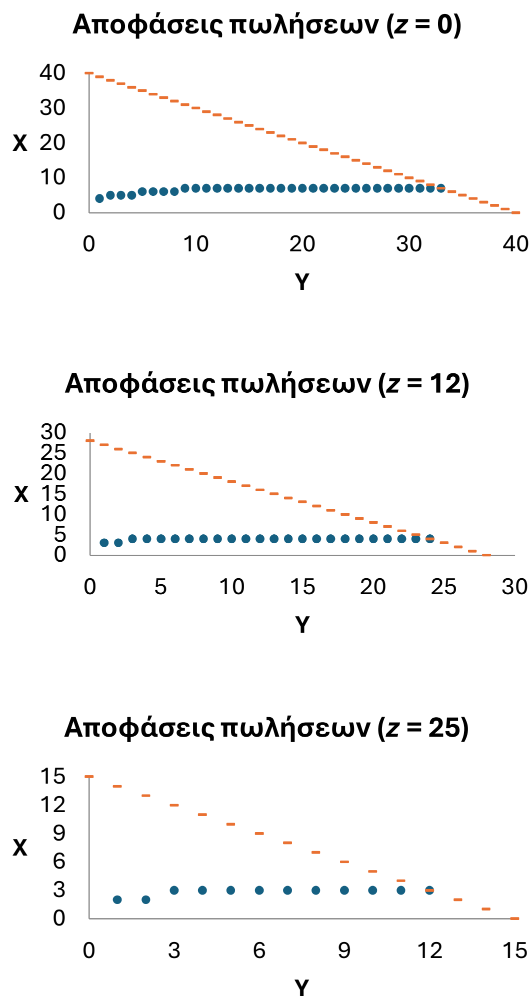
3.3 Αριθμητική διερεύνηση της βέλτιστης πολιτικής

Για να διερευνήσουμε τη δομή της βέλτιστης πολιτικής, εκτελέσαμε μια σειρά αριθμητικών πειραμάτων εφαρμόζοντας τον αλγόριθμο διαδοχικών προσεγγίσεων στην εξίσωση 3.1. Για τις αριθμητικές δοκιμές χρησιμοποιήσαμε τις ακόλουθες τιμές παραμέτρων: $M = 40$, $\lambda_1 = 1$, $\lambda_2 = 0.1$, $\mu = 40$, $p = 1$, $r = 0.8$, $b_1 = 0.2$ and $b_2 = 0.12$. Οι πιθανότητες ποιότητας και ικανοποίησης των εξερχόμενων προϊόντων που χρησιμοποιήθηκαν στα αριθμητικά πειράματα παρουσιάζονται στον Πίνακα 3.1. Αξίζει να σημειωθεί ότι ο αλγόριθμος διαδοχικών προσεγγίσεων δίδει $J^* = 9.62094$ ως τον βέλτιστο μέσο ρυθμό κέρδους για την αρχική περίπτωση που εξετάσαμε.

Πίνακας 3.1 Πιθανότητες παραγωγής και ικανοποίησης πελατών για το σύστημα υπό εξέταση.

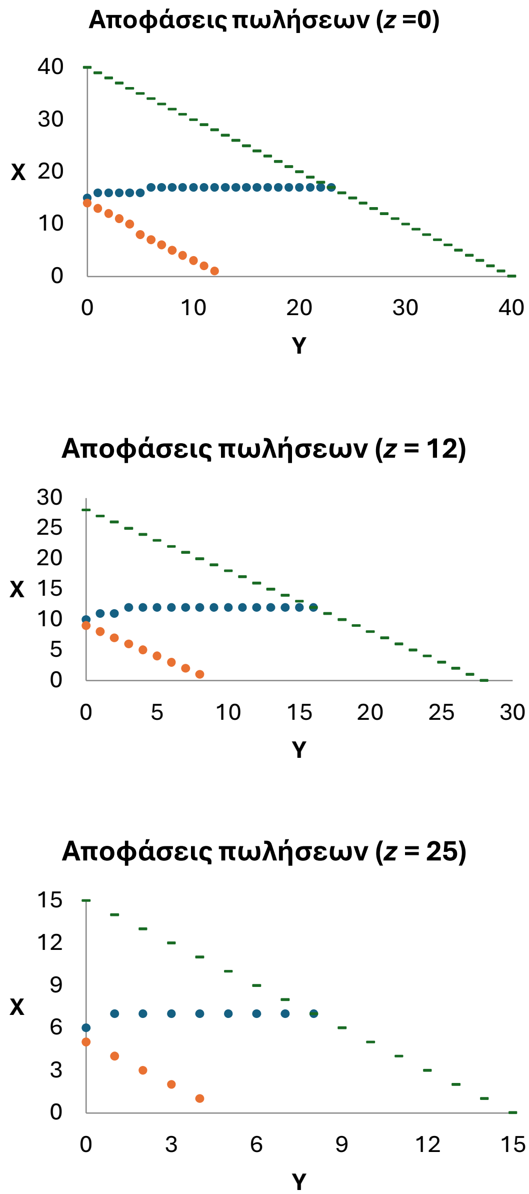
Επίπεδο ποιότητας *i*:	1	2	3
p_i	0.495	0.410	0.095
s_i^1	0.835	0.555	0.222
s_i^2	0.800	0.500	0.200

Στο Σχήμα 3.1 παρουσιάζουμε τις βέλτιστες πολιτικές για αυτό το σύστημα όταν το παραγόμενο προϊόν είναι ποιότητας επιπέδου 1 και ο αριθμός των τακτικών πελατών εκτός συστήματος, z, είναι 0, 12 ή 25. Λόγω της πολυπλοκότητας του χώρου καταστάσεων και για να έχουμε καλύτερη παρουσίαση, επιλέξαμε να παρουσιάσουμε τις βέλτιστες πολιτικές για ορισμένες χαρακτηριστικές τιμές του z. Σε αυτή την περίπτωση, η καμπύλη που χωρίζει τον τριγωνικό χώρο καταστάσεων καθορίζει το όριο μεταξύ των αποφάσεων 1 και 2, με την περιοχή των αποφάσεων 1 να βρίσκεται πάντα πάνω από τη γραμμή. Τα αποτελέσματα είναι αρκετά λογικά. Οι τακτικοί πελάτες έχουν υψηλότερα κόστη αναμονής και ρυθμούς εσόδων και, γενικά, είναι πιο σημαντικοί από τους περιστασιακούς πελάτες. Έτσι, όταν έχετε έναν αριθμό τακτικών πελατών σε αναμονή μέσα στο σύστημα, και επιπλέον έχετε ένα προϊόν υψηλής ποιότητας, είναι λογικό να δίνετε προτεραιότητα στους τακτικούς πελάτες ανεξάρτητα από τον αριθμό των περιστασιακών πελατών που περιμένουν. Καθώς αυξάνεται ο αριθμός των τακτικών πελατών σε αναμονή εκτός συστήματος παραγωγής, γίνεται πιο δύσκολο να δοθεί προτεραιότητα στους περιστασιακούς πελάτες.



Σχήμα 3.1 Βέλτιστες πολιτικές για τρία διαφορετικά επίπεδα τακτικών πελατών σε αναμονή εκτός συστήματος z, για προϊόντα ποιότητας επιπέδου 1. Η οριζόντια γραμμή είναι το όριο μεταξύ των καταστάσεων απόφασης 1 (πάνω από τη γραμμή) και των καταστάσεων απόφασης 2 (κάτω από τη γραμμή)

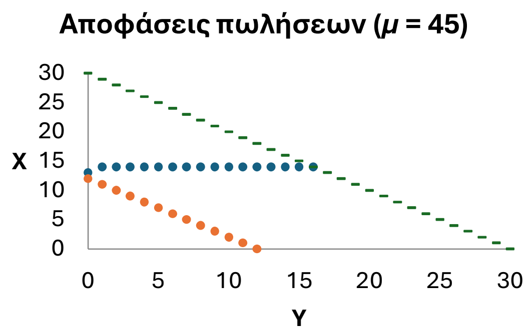
Στο Σχήμα 3.2 παρουσιάζουμε τις βέλτιστες πολιτικές για αυτό το σύστημα όταν το παραγόμενο προϊόν είναι ποιότητας επιπέδου 3 και ο αριθμός των τακτικών πελατών σε αναμονή εκτός συστήματος παραγωγής z είναι 0, 12 ή 25. Όπως μπορούμε να δούμε, σε αυτή την περίπτωση ο τριγωνικός χώρος καταστάσεων χωρίζεται σε τρεις περιοχές. Η ανώτερη περιοχή είναι η περιοχή των αποφάσεων πώλησης 1 (πώληση του προϊόντος σε τακτικό πελάτη), η κάτω δεξιά περιοχή είναι η περιοχή των αποφάσεων πώλησης 2 (πώληση του προϊόντος σε περιστασιακό πελάτη) και η κάτω αριστερή περιοχή είναι η περιοχή των αποφάσεων πώλησης 3 (απόρριψη του προϊόντος). Τα αποτελέσματα είναι ξανά αρκετά λογικά, καθώς ένας υψηλός αριθμός τακτικών πελατών μέσα στο σύστημα είναι η μόνη περίπτωση που μπορεί να αξίζει τον κόπο να πάρουμε το ρίσκο να πουλήσουμε ένα προϊόν χαμηλής ποιότητας σε τακτικό πελάτη. Επιπλέον, καθώς μειώνεται ο αριθμός των τακτικών πελατών σε αναμονή εκτός συστήματος, γίνεται λιγότερο κερδοφόρο να λάβουμε αυτήν την απόφαση

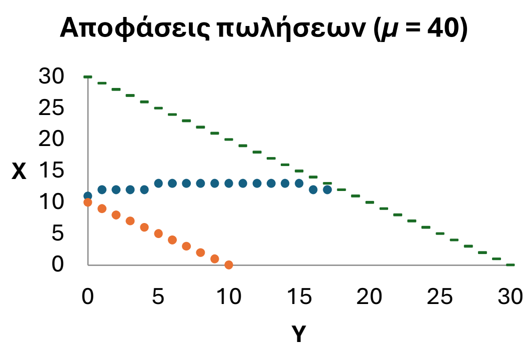


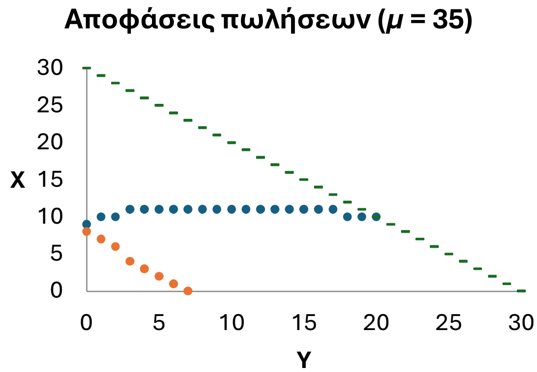
Σχήμα 3.2 Βέλτιστη πολιτική για τρία διαφορετικά επίπεδα τακτικών πελατών σε αναμονή εκτός συστήματος z, για προϊόντα ποιότητας επιπέδου 3. Στην ανώτερη

περιοχή η απόφαση 1 είναι βέλτιστη, στην κάτω δεξιά περιοχή η απόφαση 2 είναι βέλτιστη και στην κάτω αριστερή περιοχή η απόφαση 3 είναι βέλτιστη.

Στη συνέχεια, εξετάζουμε την ευαισθησία της βέλτιστης πολιτικής στις μεταβολές του μέσου ρυθμού παραγωγής μ. Στο Σχήμα3 παρουσιάζουμε τις βέλτιστες αποφάσεις που προέκυψαν για τρεις διαφορετικές τιμές του μ, όταν ο αριθμός των τακτικών πελατών σε αναμονή εκτός συστήματος παραγωγής είναι $z = 10$ και το παραγόμενο προϊόν είναι ποιότητας επιπέδου 3. Μπορούμε να δούμε ότι καθώς αυξάνεται ο μέσος ρυθμός παραγωγής, γίνεται ευκολότερο να απορρίψουμε ένα προϊόν χαμηλής ποιότητας και να αποφύγουμε την πώλησή του σε τακτικό πελάτη.

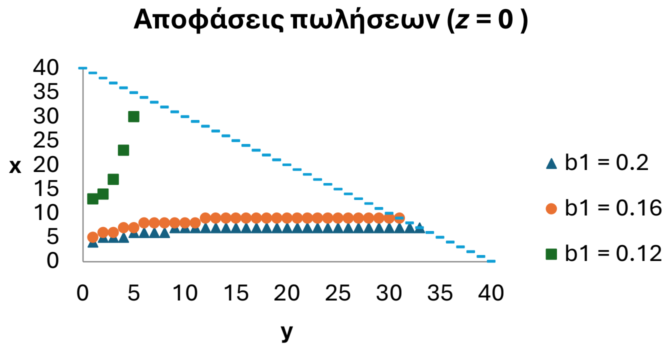






Σχήμα 3.3 Βέλτιστες πολιτικές για διαφορετικά επίπεδα ρυθμού παραγωγής μ, όταν z = 10 και το προϊόν είναι ποιότητας επιπέδου 3. Στην ανώτερη περιοχή η απόφαση 1 είναι βέλτιστη, στην κάτω δεξιά η απόφαση 2 είναι βέλτιστη και στην κάτω αριστερή η απόφαση 3.

Τέλος, εξετάζουμε πόσο ευαίσθητες είναι οι βέλτιστες πολιτικές στις μεταβολές του μοναδιαίου κόστους αναμονής για τους τακτικούς πελάτες b_1. Στο Σχήμα 3.4 παρουσιάζουμε τις βέλτιστες αποφάσεις που προκύπτουν για τρεις διαφορετικές τιμές του b_1, για αριθμό τακτικών πελατών σε αναμονή εκτός συστήματος $z = 0$ και για παραγόμενα προϊόντα ποιότητας επιπέδου 1. Οι καμπύλες στο διάγραμμα δείχνουν το όριο μεταξύ των περιοχών αποφάσεων 1 και 2, σε κάθε περίπτωση δοκιμής. Στην περιοχή πάνω από τις καμπύλες οι αποφάσεις 1 είναι βέλτιστες, ενώ στην περιοχή κάτω από τις καμπύλες οι αποφάσεις 2 είναι βέλτιστες. Πρέπει να αναφέρουμε εδώ ότι το μοναδιαίο κόστους αναμονής για τους περιστασιακούς πελάτες b_2 είναι 0.12 σε όλες τις τρεις περιπτώσεις μελέτης που παρουσιάζονται. Εφόσον οι τακτικοί πελάτες έχουν υψηλότερο μοναδιαίου κόστους αναμονής από τους περιστασιακούς πελάτες, είναι βέλτιστο να πωλείται ένα προϊόν υψηλής ποιότητας σε τακτικό πελάτη, όταν υπάρχουν αρκετοί τακτικοί πελάτες σε αναμονή, ανεξάρτητα από τον αριθμό των περιστασιακών πελατών που περιμένουν εξυπηρέτηση. Στην περίπτωση όπου θεωρούμε ότι το μοναδιαίο κόστους αναμονής είναι ίδιο και για τις δύο κατηγορίες πελατών, βλέπουμε ότι η βέλτιστη πολιτική αλλάζει πολύ και δεν υπάρχει πραγματική προτεραιότητα για τους τακτικούς πελάτες.



Σχήμα 3.4 Βέλτιστες πολιτικές για διαφορετικές τιμές του μοναδιαίου κόστους αναμονής b_1, όταν z = 0 και το προϊόν είναι ποιότητας επιπέδου 1. Η απόφαση 1 είναι βέλτιστη στην περιοχή πάνω από την καμπύλη και η απόφαση 2 είναι βέλτιστη κάτω από την καμπύλη.

3.4 Ευρετική πολιτική τύπου κατωφλίου

Όπως και στα συστήματα του προηγούμενου κεφαλαίου, η αριθμητική διερεύνηση της βέλτιστης πολιτικής ελέγχου ποιότητας μας δείχνει ότι αυτή έχει ιδιαίτερα σύνθετη δομή. Ο λόγος που συμβαίνει αυτό είναι ότι η βέλτιστη πολιτική καθορίζεται τόσο από το επίπεδο ποιότητας των προϊόντων, όσο και από το διάνυσμα κατάστασης. Εδώ η πολυπλοκότητα αυξάνει καθώς οι πελάτες σε αναμονή για ικανοποίηση της παραγγελίας τους διατηρούν την γνώση της κατηγορία τους, άρα έχουμε περισσότερες μεταβλητές κατάστασης και αποφάσεις που πρέπει να πάρουμε. Ως αποτέλεσμα η βέλτιστη πολιτική είναι πρακτικά μη εφαρμόσιμη καθώς οι ανθρώπινοι χειριστές προτιμούν την χρήση απλών κανόνων και όχι περίπλοκες πολιτικές που παράγονται από υπολογιστές. Παρ'όλα αυτά, υπάρχουν ορισμένες τιμές του i για τις οποίες η βέλτιστη πολιτική ποιότητας είναι ανεξάρτητη από την κατάσταση του συστήματος. Στο προηγούμενο παράδειγμα, πάντα πωλούμε προϊόντα με επίπεδα ποιότητας $i = 1, 2$. Η μόνη εξαίρεση είναι όταν $i = 3$, όπου και οι δύο αποφάσεις είναι εφαρμόσιμες και η βέλτιστη πολιτική καθορίζεται από καμπύλες εναλλαγής αποφάσεων, που χωρίζουν τον χώρο καταστάσεων σε συμπαγείς περιοχές αποφάσεων όπως φαίνεται στα Σχήματα 3.2–3.4.

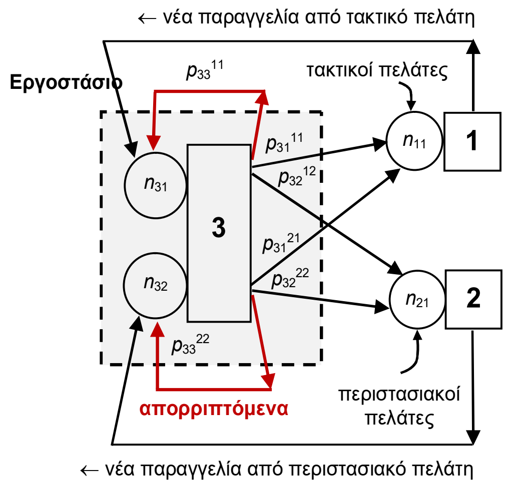
Σε αυτή την ενότητα, εξετάζουμε μια εύχρηστη ευρετική πολιτική τύπου κατωφλίου, ως εναλλακτική της βέλτιστης πολιτικής. Υποθέτουμε ότι το Q είναι ένα κατώφλι ποιότητας ($Q \leq I$), έτσι ώστε όλα τα αντικείμενα με $i > Q$ να απορρίπτονται, διαφορετικά αποστέλλονται στους αναμένοντες πελάτες. Στη συνέχεια, το πρόβλημα είναι η εύρεση του Q που μεγιστοποιεί τον μέσο ρυθμό κέρδους. Το κύριο πλεονέκτημα της προτεινόμενης πολιτικής είναι η υπολογιστική της αποδοτικότητα. Για περίπλοκα δίκτυα παραγωγής, η βέλτιστη πολιτική δεν μπορεί να υπολογιστεί λόγω της έκρηξης του χώρου καταστάσεων. Η εύρεση της βέλτιστης πολιτικής είναι απαιτητική υπολογιστικά ακόμα και για συστήματα ενός σταδίου. Από την άλλη πλευρά, η ευρετική πολιτική απαιτεί λιγότερους υπολογισμούς και μπορεί να εφαρμοστεί σε συστήματα παραγωγής με αυξημένη πολυπλοκότητα, όπως γραμμές και δίκτυα παραγωγής με πολλούς σταθμούς εργασίας. Όπως είδαμε στις προηγούμενες παραγράφους η βέλτιστη πολιτική δίνει προτεραιότητα στην ικανοποίηση των τακτικών πελατών και μόνο υπό συγκεριμένες συνθήκες τα παραγόμενα προϊόντα πωλούνται στους περιστασιακούς πελάτες. Η προτεινόμενη πολιτική κατωφλίου από την άλλη δεν διακρίνει ανάμεσα στις δύο κατηγορίες πελατών. Η συγκεκριμένη προσέγγιση μπορεί να είναι υποβέλτιστη αλλά είναι εφαρμόσιμη ακόμη και όταν η εταιρεία παραγωγής έχει μόνο μερική πληροφόρηση για την κατάσταση της αγοράς, π.χ., όταν είναι γνωστό το M αλλά τα μερίδια αγοράς z και $M - x - y - z$ που αντιστοιχούν σε τακτικούς και περιστασιακούς πελάτες είναι άγνωστα.

3.4.1 Εκτίμηση μέτρων απόδοσης

Όταν το σύστημα ενός σταδίου λειτουργεί υπό την προτεινόμενη πολιτική ελέγχου ποιότητας, μπορεί να μοντελοποιηθεί ως κλειστό δίκτυο ουρών αναμονής με δύο κατηγορίες πελατών (ΚΔΑ$_2$) με $N = 3$ κόμβους και M εργασίες. Ο κόμβος $j = 3$ του ΚΔΑ$_2$ έχει ένα μόνο εκθετικό εξυπηρετητή με μέσο ρυθμό εξυπηρέτησης μ ίσο με τον μέσο ρυθμό παραγωγής της μονάδας παραγωγής. Οι κόμβοι 1 και 2 αντιπροσωπεύουν αντίστοιχα τους μηχανισμούς που παράγουν αφίξεις τακτικών και περιστασιακών πελατών και σε κάθε χρονική στιγμή έχουν τόσους ενεργούς εξυπηρετητές όσες είναι οι εργασίες στις ουρές τους. Οι εξυπηρετητές σε αυτούς τους δύο κόμβους έχουν εκθετικά κατανεμημένους χρόνους επεξεργασίας με μέσες τιμές $1/\lambda_j$, για $j = 1, 2$. Το ΚΔΑ$_2$ και το σύστημα παραγωγής έχουν την ίδια σχηματική αναπαράσταση όπως φαίνεται στο Σχήμα 3.5.

Σε οποιαδήποτε χρονική στιγμή, το διάνυσμα κατάστασης του συστήματος παραγωγής είναι ίδιο με το διάνυσμα του πλήθους αριθμού εργασιών, κάθε κατηγορίας πελατών, στους κόμβους του ΚΔΑ$_2$. Πιο συγκεκριμένα n_{jk} είναι το πλήθος πελατών τύπου

k στον κόμβο j, ενώ p_{jm}^{kl} είναι οι πιθανότητες μετάβασης από τον κόμβο j στον κόμβο m, και από την κατηγορία πελατών k στην κατηγορία l. Για παράδειγμα n_{31} είναι το πλήθος των τακτικών πελατών που αναμένουν ικανοποίηση των παραγγελιών τους από το σύστημα παραγωγής, n_{32} είναι το πλήθος των περιστασιακών πελατών που αναμένουν εντός του συστήματος παραγωγής, ενώ n_{11} και n_{22} είναι οι τακτικοί και περιστασιακοί πελάτες αντίστοιχα σε τροχιά εκτός συστήματος παραγωγής. Όπως γίνεται εύκολα αντιληπτό στο σύστημα μας οι κόμβοι 1 και 2 εξυπηρετούν μόνο πελάτες της κατηγορίας 1 και 2 αντίστοιχα δηλαδή έχουμε ότι $n_{12} = n_{21} = 0$. Οι πιθανότητες δρομολόγησης του κόμβου 3 εξαρτώνται από την ποιότητα του αντίστοιχου εξερχόμενου προϊόντος, την απόφαση ποιοτικού ελέγχου που λαμβάνεται στο σύστημα παραγωγής και την κατηγορία των πελατών. Υπό την προτεινόμενη πολιτική, η ποιότητα του τελικού προϊόντος i προσδιορίζεται με επιθεώρηση και αν $i > Q$ (χαμηλή ποιότητα), τότε αυτό το αντικείμενο απορρίπτεται και μια νέα πρώτη ύλη απελευθερώνεται στην πρώτη μηχανή της μονάδας παραγωγής. Στην ουσία, αυτό είναι το ίδιο με μια εργασία που δρομολογείται από τον κόμβο 3 πίσω στην είσοδο του κόμβου 3, όπως φαίνεται στο Σχήμα 3.5. Οι αντίστοιχες πιθανότητες δρομολόγησης δίδονται από την εξίσωση $p_{33}^{kk} = p_{33} = p_{Q+1} + \ldots + p_I$, $k = 1, 2$. Αν το τελικό προϊόν έχει αποδεκτό επίπεδο ποιότητας ($i \leq Q$), τότε θα αγοραστεί από έναν πελάτη. Σε αυτό το κεφάλαιο η ικανοποίηση ενός πελάτη εξαρτάται και από την κατηγορία στην οποία ανήκει συνεπώς με πιθανότητα $p_{31}^{k1} = p_1 s_1^k + \ldots + p_I s_Q^k$, $k = 1, 2$, ένας πελάτης κατηγορίας k, που αγοράζει ένα προϊόν θα είναι ικανοποιημένος. Στο ισοδύναμο ΚΔΑ$_2$, μια εργασία αναχωρεί από τον κόμβο 3 και δρομολογείται στον κόμβο 1 (ο ικανοποιημένος πελάτης γίνεται τακτικός πελάτης). Ωστόσο, με πιθανότητα $p_{32}^{k2} = p_1(1 - s_1^k) + \ldots + p_I(1 - s_Q^k)$, $k = 1, 2$, ο πελάτης θα είναι δυσαρεστημένος και στο ισοδύναμο ΚΔΑ$_2$ η εργασία θα πάει στον κόμβο 2 (ο δυσαρεστημένος πελάτης γίνεται περιστασιακός πελάτης). Τέλος, όλες οι εργασίες που προέρχονται από τους κόμβους 1 και 2 δρομολογούνται στον κόμβο 3 του ΚΔΑ, δηλαδή, $p_{13}^{11} = p_{23}^{22} = 1$.



Σχήμα 3.5 Σύστημα παραγωγής κατά παραγγελία ενός σταδίου με δύο κατηγορίες πελατών που έχουν μνήμη της κατάστασης τους.

Στη συνέχεια περιγράφουμε έναν αλγόριθμο από τη θεωρία ουρών αναμονής, ο οποίος επιτρέπει τον υπολογισμό των πιθανοτήτων μόνιμης κατάστασης $P(n) = P(n_{11}, n_{12}, n_{21}, n_{22}, n_{31}, n_{32})$ και όλων των συνιστωσών του μέσου ρυθμού κέρδους του συστήματος, όταν χρησιμοποιείται η προτεινόμενη ευρετική πολιτική.

Ας ονομάσουμε TH_3 το μέσο ρυθμό παραγωγής του κόμβου 3, $B_k = E(n_{3k})$ το μέσο έλλειμμα των πελατών τύπου k στον κόμβο 3, που είναι ίσο με το μέσο πλήθος εκκρεμών παραγγελιών πελατών τύπου k, $\Pi = [p_{jm}^{kl}]$, $j,m = 1, 2, 3$, $k, l = 1, 2$, τον πίνακα πιθανοτήτων δρομολόγησης και $u = [u_1^1, \ldots, u_3^2]$ οποιαδήποτε μη αρνητική λύση του συστήματος των γραμμικών εξισώσεων $u = u\Pi$. Το διάνυσμα u καθορίζεται ως γινόμενο μιας σταθεράς, έτσι είμαστε ελεύθεροι να επιλέξουμε ένα σχήμα κανονικοποίησης για τα u_i (π.χ. $u_3^1 = 1$ ή $u_3^1 + u_3^2 = 1$). Ας υποθέσουμε ότι $\alpha_j(n_j)$ είναι ο αριθμός των κατειλημμένων εξυπηρετητών στον κόμβο j, όπου n_j είναι το πλήθος εργασιών όλων των κατηγοριών στον κόμβο j, δηλαδή $n_j = n_{j1} + n_{j2}$. Έχουμε ότι $\alpha_j(n_j) = 1$ για τον κόμβο $j = 3$, για κάθε $n_3 \geq 1$, δεδομένου ότι έχει μόνο έναν εξυπηρετητή, ενώ για τους κόμβους 1 και 2 έχουμε ότι $\alpha_1(n_1) = n_1$ και $\alpha_2(n_2) = n_2$. Στη συνέχεια, υπολογίζουμε αναδρομικά τις ακολουθίες $\beta_j(n_j) = \alpha_j(n_j)\beta_j(n_j - 1)$ χρησιμοποιώντας την $\beta_j(0) = 1$ ως οριακή τιμή. Αποδεικνύεται ότι $\beta_j(n_j) = 1$ για $j = 3$ και $\beta_j(n_j) = n_j!$ για $j = 1, 2$. Οι πιθανότητες μόνιμης κατάστασης του συστήματος δίνονται από (βλ. π.χ. Παράγραφο 8.6.1 στο βιβλίο των

Buzacott and Shanthikumar [5]).

$$P(n) = \frac{1}{G(M)}\left[\prod_{j=1}^{N}\left(\frac{n_j!}{n_{j1}!\,n_{j2}!}\right)\prod_{k=1}^{K}\left(u_j^k\right)^{n_{jk}} \Big/ \prod_{l=1}^{n_j}\mu_j\ \alpha_j(n_j)\right]$$

$$= \frac{1}{G(M)}\left[\prod_{j=1}^{N}\left(\frac{n_j!}{n_{j1}!\,n_{j2}!}\right)\frac{1}{\beta_j(n_j)}\prod_{k=1}^{K}\left(\frac{u_j^k}{\mu_j}\right)^{n_{jk}}\right]$$

όπου $\mu_1 = \lambda_1$, $\mu_2 = \lambda_2$, $\mu_3 = \mu$, και $G(M)$ είναι μια σταθερά κανονικοποίησης που προκύπτει από την σχέση

$$G(M) = \sum_{n_{11}+\cdots+n_{32}=M}\left[\prod_{j=1}^{N}\left(\frac{n_j!}{n_{j1}!\,n_{j2}!}\right)\left[\frac{\prod_{k=1}^{2}\left(u_j^k\right)^{n_{jk}}}{\prod_{l=1}^{n_j}\mu_j\alpha_j(n_j)}\right]\right]$$

Αν συμπτύξουμε τους πελάτες όλων των κατηγοριών σε κάθε κόμβο του δικτύου και θέσουμε $\underline{n} = (n_1, n_2, n_3)$, αποδεικνύεται (βλ.[5]) ότι οι πιθανότητες $P(\underline{n})$ προκύπτουν από την εξίσωση

$$P(\underline{n}) = \frac{1}{\underline{G}(M)}\left[\prod_{j=1}^{N}\frac{1}{\beta_j(n_j)}\left(\frac{u_j}{\mu_j}\right)^{n_j}\right],$$

όπου $u_j = u_j^1 + \ldots + u_j^K$, $K = 2$, ενώ η σταθερά κανονικοποίησης δίδεται από

$$\underline{G}(M) = \sum_{n_1+\cdots+n_N=M}\left[\prod_{j=1}^{N}\frac{1}{\beta_j(n_j)}\left(\frac{u_j}{\mu_j}\right)^{n_j}\right].$$

Στην πραγματικότητα αν συγκρίνουμε τις παραπάνω εξισώσεις με τις εξισώσεις (2.3) και (2.4) θα δούμε ότι είναι ίδιες με τις πιθανότητες μόνιμης κατάστασης ενός ΚΔΑ με έναν τύπο πελατών (βλ.[5]).

Τα μέτρα απόδοσης του συστήματος υπολογίζονται από γνωστούς τύπους των συστημάτων αναμονής (βλ.[5]). Ο μέσος ρυθμός παραγωγής του κόμβου $j = 3$ (δηλαδή της μονάδας παραγωγής) είναι

$$\mathrm{TH}_3 = \sum_{n_3=1}^{M}\mu P(n_3) = \mu[1 - P(n_3 = 0)] = \mu\left[1 - \sum_{n_1+n_2=M}P(n_1, n_2, 0{,}0)\right]$$

$$= \rho_3\frac{\underline{G}(M-1)}{\underline{G}(M)}$$

όπου $\rho_j = u_j/\mu_j$. Ο τελευταίος όρος της εξίσωσης προκύπτει από την συμπτυξη του $ΚΔΑ_2$

σε ένα δίκτυο με έναν τύπο πελατών και έχει την ίδια μορφή με την εξίσωση 2.5. Ο μέσος αριθμός εκκρεμών παραγγελιών της κατηγορίας k είναι

$$B_k = E(n_{3k}) = \sum_{n_{3k}=0}^{M} n_{3k} \sum_{n_{11}+n_{22}+n_{3\bar{k}}=M-n_{3k}} P(n_{11}, n_{22}, n_{31}, n_{32})$$

όπου $\bar{k}$=2, όταν k =1 και $\bar{k}$=1 όταν k = 2. Ο συνολικός μέσος ρυθμός κέρδους του συστήματος είναι ίσος με

$$J = p(1 - p_{33})\mathrm{TH}_3 - rp_{33}\mathrm{TH}_3 - \sum_{k=1}^{2} b_k B_k$$

Η τιμή της συνάρτησης μέσου ρυθμού κέρδους της ευρετικής πολιτικής J εξαρτάται από την επιλογή του επιπέδου ποιότητας Q = 1, …, I, που καθορίζει τις πιθανότητες δρομολόγησης p_{33}. Για να βρούμε το βέλτιστο Q πραγματοποιούμε εξαντλητική αναζήτηση, με τρόπο όμοιο με αυτό που εφαρμόσαμε στην περίπτωση της ευρετικής πολιτικής του προηγούμενου κεφαλάιου.

1) Αρχικοποιούμε την βέλτιστη πολιτική, ορίζοντας $J^* = -\infty$ και το όριο ποιότητας Q = 0.

2) Για Q = 1, …, I:

α) Υπολογίζουμε τις αντίστοιχες πιθανότητες δρομολόγησης $p_{31}^{11}, p_{31}^{12}, p_{31}^{21}, p_{31}^{22}$ και p_{33}, καθώς και τις πιθανότητες μόνιμης κατάστασης $P(n)$ του ΚΔΑ$_2$ και τον συνολικό μέσο ρυθμό κέρδους J του συστήματος.

β) Αν $J^* > J$, η τρέχουσα Q είναι η τοπικά βέλτιστη απόφαση: αποθηκεύουμε ορίζουμε $J^* = J$ και $Q^* = Q$.

3) Επαναλαμβάνουμε τα παραπάνω βήματα μέχρι να εξετασθούν όλα τα επίπεδα ποιότητας $Q \leq I$. Το βέλτιστο ζεύγος είναι (Q^*, J^*).

3.5 Αριθμητική σύγκριση της ευρετικής με την βέλτιστη πολιτική

Σε αυτή την ενότητα, συγκρίνουμε αριθμητικά την προτεινόμενη πολιτική με τη βέλτιστη πολιτική, για να δούμε αν η πρώτη μπορεί να χρησιμοποιηθεί ως μια εύκολα υλοποιήσιμη εναλλακτική της δεύτερης με ανεκτές αποκλίσεις στην απόδοση.

Εξετάζουμε ένα σύστημα παραγωγής ενός σταδίου, όπως αυτό που περιγράφεται στις προηγούμενες παραγράφους. Υποθέτουμε ότι M = 20 και τις ίδιες πιθανότητες επιπέδου ποιότητας p_q και πιθανότητες ικανοποίησης s_q όπως στην Παράγραφο 3.2 (Πίνακας 3.1). Ο Πίνακας 3.2 δείχνει τις τιμές των υπόλοιπων παραμέτρων και τις αντίστοιχες βέλτιστες πολιτικές.

Βλέπουμε ότι οι μέσοι ρυθμοί κέρδους των δύο πολιτικών είναι ίσοι έως δύο δεκαδικά ψηφία. Η μέγιστη παρατηρούμενη σχετική απόκλιση είναι μικρότερη από 0.15%. Έτσι, για τα σύνολα δεδομένων που χρησιμοποιήθηκαν σε αυτή τη μελέτη, η προτεινόμενη πολιτική φαίνεται να είναι μια εξαιρετική προσέγγιση της βέλτιστης πολιτικής. Για τις συγκεκριμένες τιμές παραμέτρων φαίνεται, ότι το όφελος που προκύπτει από την ευελιξία που μας δίνει η βέλτιστη πολιτική, σχετικά με το τι προϊόντα θα πουλήσουμε σε ποιους πελάτες, δεν είναι τόσο σημαντικό. Γενικά σε όσα αριθμητικά πειράματα πραγματοποιήσαμε δεν παρατηρήσαμε σημαντικές αποκλίσεις ανάμεσα στις δύο πολιτικές. Όπως και στο Κεφάλαιο 2 η ευρετική πολιτική εμφανίζεται ως ένα εξαιρετικά αποδοτικό και εύχρηστο υποκατάστατο της βέλτιστης πολιτικής.

Πίνακας 3.2 Σύγκριση πολιτικών για σύστημα ενός σταδίου και διάφορες τιμές των παραμέτρων του

Τιμές παραμέτρων							**Βέλτιστη πολιτική**	**Πολιτική κατωφλίου**	
λ_1	λ_2	μ	c	r	B_1	B_2	**Μέσο κέρδος**	**Μέσο κέρδος**	**Κατώφλι ποιότητας**
1	0.1	25	0.8	1	0.2	0.12	4.769	4.763	επίπεδο 2
0.8	0.1	25	0.8	1	0.2	0.12	4.561	4.557	επίπεδο 2
1	0.1	25	0.8	1	0.2	0.12	4.849	4.843	επίπεδο 2
1	0.07	25	0.8	1	0.2	0.12	3.553	3.549	επίπεδο 2
1	0.2	25	0.8	1	0.2	0.12	8.032	8.022	επίπεδο 3
1	0.1	18	0.8	1	0.2	0.12	4.704	4.697	επίπεδο 2
1	0.1	20	0.8	1	0.2	0.12	4.729	4.722	επίπεδο 2
1	0.1	25	0.6	1	0.2	0.12	4.879	4.874	επίπεδο 2
1	0.1	25	0.9	1	0.2	0.12	4.714	4.708	επίπεδο 2
1	0.1	25	0.8	0.9	0.2	0.12	4.243	4.238	επίπεδο 2
1	0.1	25	0.8	1.1	0.2	0.12	5.295	5.289	επίπεδο 2
1	0.1	25	0.8	1	0.18	0.12	4.773	4.768	επίπεδο 2
1	0.1	25	0.8	1	0.22	0.12	4.765	4.759	επίπεδο 2
1	0.1	25	0.8	1	0.2	0.1	4.770	4.765	επίπεδο 2
1	0.1	25	0.8	1	0.2	0.15	4.766	4.761	επίπεδο 2

3.6 Σύνοψη

Σε αυτό το κεφάλαιο παρουσιάσαμε ένα μοντέλο, για ένα σύστημα παραγωγής κατά παραγγελίες (make-to-order), το οποίο ενσωματώνει την ικανοποίηση των πελατών και τη δυναμική της αγοράς στις αποφάσεις ελέγχου ποιότητας. Το μοντέλο μπορεί να χρησιμεύσει ως εργαλείο για τη διερεύνηση βέλτιστων πολιτικών για τέτοια συστήματα παραγωγής. Επιπλέον, υπογραμμίζει τις αλληλεπιδράσεις μεταξύ παραγωγής, πωλήσεων και ποιότητας, δείχνοντας τις επιπτώσεις της ποιότητας στην ικανοποίηση των πελατών και στην κερδοφορία του συστήματος και υπογραμμίζει τη σημασία της γνώσης των προηγούμενων καταστάσεων των πελατών και τον τρόπο με τον οποίο αυτό επηρεάζει την βέλτιστη πολιτική.

Ακόμη παρουσιάσαμε μια απλή και αποδοτική ευρετική πολιτική. Για την ανάλυση της προτεινόμενης πολιτικής χρησιμοποιήσαμε αποτελέσματα από την θεωρία κλειστών δικτύων αναμονής, που εξυπηρετούν πολλές κατηγορίες πελατών. Η πολιτική αυτή προσεγγίζει ικανοποιητικά την απόδοση της βέλτιστης πολιτικής, ενώ είναι υπολογιστικά οικονομική και συγχρόνως εύχρηστη λόγω της απλότητας της.

Μέχρι τώρα έχουμε εξετάσει συτήματα παραγωγής κατά παραγγελίες (make-to-order). Στο Κεφάλαιο 4 θα δούμε συστήματα παραγωγής προς αποθεματοποίηση (make-to-stock).

Μια άμεση επέκταση, στο πρόβλημα που μελετήσαμε σε αυτό το κεφάλαιο, θα μπορούσε να είναι η εξέταση σύνθετων συστημάτων παραγωγής (δίκτυα ή γραμμές παραγωγής με πολλούς σταθμούς εργασίας). Επίσης σχετικά εύκολα θα μπορούσαμε να επεκτείνουμε την ανάλυση μας σε συστήματα με πολλές κατηγορίες πελατών. Με αυτό τον τρόπο θα μπορούσαμε να μελετήσουμε σύνθετες καταναλωτικές συμπεριφορές που δεν περιγράφονται ικανοποιητικά από την μοντελοποίηση με τις δύο κατηγορίες πελατών. Η ύπαρξη κατάλληλων μοντέλων κλειστών δικτύων αναμονής με πολλές κατηγορίες πελατών φαίνεται να διευκολύνει την έρευνα προς αυτή την κατεύθυνση. Η προσθήκη και του χρονικού παράγοντα στην ικανοποίηση πελατών από την άλλη πλευρά φαίνεται ιδιαίτερα δύσκολη υπόθεση και μάλλον θα πρέπει να καταφύγουμε σε προσεγγιστικές αλλά ιδιαίτερα επιτυχημένες προσεγγίσεις όπως αυτές της προσομοίωσης και της ενισχυτικής μάθησης.

4 Επιπτώσεις της ποιότητας στο μερίδιο αγοράς και στην κερδοφορία απλών συστημάτων παραγωγής προς αποθήκευση

4.1 Εισαγωγή

Σε αυτό το κεφάλαιο μελετάμε ένα Μαρκοβιανό, σύστημα ενός σταδίου που αντιμετωπίζει τυχαία ζήτηση. Οποιαδήποτε παραγγελία δεν ικανοποιείται αμέσως από το απόθεμα χάνεται προς τους ανταγωνιστές. Υποθέτουμε ότι η αγορά είναι πεπερασμένη και αποτελείται από τακτικούς και περιστασιακούς πελάτες. Οι τακτικοί πελάτες έχουν υψηλότερους μέσους ρυθμούς ζήτησης από τους περιστασιακούς πελάτες. Κάθε εξερχόμενο προϊόν επιθεωρείται και ταξινομείται ως υψηλής ποιότητας, μεσαίας ποιότητας ή μη συμμορφούμενο. Ο πελάτης που αγοράζει ένα αντικείμενο εντάσσεται στην τάξη των τακτικών ή περιστασιακών πελατών, με τις αντίστοιχες πιθανότητες να εξαρτώνται από το επίπεδο ποιότητας και την προηγούμενη κατάσταση του πελάτη. Όσο υψηλότερο είναι το επίπεδο ποιότητας, τόσο μεγαλύτερη είναι η πιθανότητα ένας πελάτης να παραμείνει ή να γίνει τακτικός πελάτης. Στόχος μας είναι να διερευνήσουμε τη δομή της βέλτιστης πολιτικής παραγωγής, ικανοποίησης παραγγελιών και ελέγχου ποιότητας για να μεγιστοποιήσουμε τον μέσο ρυθμό κέρδους του συστήματος. Διερευνούμε αριθμητικά τη δομή της βέλτιστης πολιτικής χρησιμοποιώντας στοχαστικό δυναμικό προγραμματισμό.

Στην Παράγραφο 4.2, περιγράφουμε ένα σύστημα ενός σταδίου, που ακολουθεί την προσέγγιση παραγωγής προς αποθεματοποίηση (make-to-stock), του οποίου η αγορά αποτελείται από τακτικούς και περιστασιακούς πελάτες. Το πρόβλημα περιγράφεται ως ένα πρόβλημα στοχαστικού δυναμικού προγραμματισμού. Μια αριθμητική διερεύνηση της βέλτιστης πολιτικής παρουσιάζεται στην Παράγραφο 4.3. Τα αποτελέσματα των αριθμητικών πειραμάτων δείχνουν και στην περίπτωση αυτή η δομή της βέλτιστης πολιτικής είναι περίπλοκη και απαιτείται ακριβή γνώση της τρέχουσας κατάστασης της αγοράς, δηλαδή του πλήθους των τακτικών και περιστασιακών πελατών αλλά και του πλήθους εκκρεμών παραγγελιών κάθε κατηγορίας. Στην Παράγραφο 4.4 παρουσιάζονται τα συμπεράσματα και πιθανές επεκτάσεις.

4.2 Περιγραφή του συστήματος

Θεωρούμε μια μονάδα παραγωγής ενός σταδίου, που παράγει ένα μόνο προϊόν, το οποίο καλύπτει τις εισερχόμενες παραγγελίες από ένα απόθεμα ήδη παραγόμενων αντικειμένων (σύστημα παραγωγής προς αποθεματοποίηση make-to-stock). Κάθε εξερχόμενο προϊόν έχει ένα επίπεδο ποιότητας i (i = 1, 2, 3 με την ποιότητα να μειώνεται όσο αυξάνεται το i) με την αντίστοιχη πιθανότητα p_i. Αν ένα εξερχόμενο προϊόν είναι ποιότητας επιπέδου 3, τότε απορρίπτεται αυτόματα, αλλιώς το διατηρούμε στο απόθεμα μαζί με προϊόντα του ίδιου επιπέδου ποιότητας. Υποθέτουμε ότι το μέγεθος της αγοράς M είναι γνωστό και σταθερό. Οι M πελάτες χωρίζονται σε δύο διακριτές κατηγορίες: την κατηγορία των τακτικών πελατών, που σημειώνεται με k = 1, και την κατηγορία των περιστασιακών πελατών, k = 2. Η κατάσταση του συστήματος μπορεί να περιγραφεί από τρεις μεταβλητές:

x στάθμη αποθέματος προϊόντων ποιότητας επιπέδου 1,

y στάθμη αποθέματος προϊόντων ποιότητας επιπέδου 2,

z πλήθος τακτικών πελατών εκτός συστήματος.

Οι αφίξεις των πελατών είναι τυχαίες και ακολουθούν την κατανομή Poisson ενώ κάθε πελάτης ζητάει μία μονάδα προϊόντος. Κάθε πελάτης έχει ένα μέσο ρυθμό ζήτησης λ_k, που εξαρτάται από την κατηγορία, k = 1, 2, όπου $\lambda_1 > \lambda_2$. Οι χρόνοι επεξεργασίας των αντικειμένων είναι ανεξάρτητες τυχαίες μεταβλητές εκθετικά κατανεμημένες με μέση τιμή $1/\mu$. Αν ένα προϊόν ποιότητας επιπέδου i πωληθεί σε έναν πελάτη της κατηγορίας k, τότε αυτός ο πελάτης θα είναι τελικά ικανοποιημένος με (συνολική) πιθανότητα s_i^k ή δυσαρεστημένος με την συμπληρωματική πιθανότητα. Οι ικανοποιημένοι πελάτες εντάσσονται στην κατηγορία των τακτικών πελατών ενώ οι δυσαρεστημένοι γίνονται περιστασιακοί πελάτες. Υποθέτουμε ότι τα προϊόντα υψηλής ποιότητας (i = 1) αντιστοιχούν σε υψηλότερες πιθανότητες ικανοποίησης, δηλαδή $s_1^k > s_2^k$ για κάθε k = 1, 2.

Το συνολικό μέτρο απόδοσης του συστήματος είναι ο μέσος ρυθμός κέρδους. Αυτό το μέγεθος εξαρτάται από τον μέσο ρυθμό πωλήσεων (που ισούται με τον μέσο ρυθμό παραγωγής του συστήματος), το κόστος των απορριπτόμενων προϊόντων και το κόστος διατήρησης αποθεμάτων. Εξετάζουμε τρεις οικονομικές παραμέτρους:

r κέρδος από την πώληση μιας μονάδας προϊόντος,

c μοναδιαίο κόστος απόρριψης (κόστος από την απόρριψη ή/και επανακατεργασία ενός προϊόντος),

h μοναδιαίο κόστος αποθέματος.

Η κατάσταση του συστήματος περιγράφεται από το διάνυσμα μεταβλητών (x, y, z) με χώρο καταστάσεων $S = \{(x, y, z) \mid x = 0, 1, \ldots, K, y = 0, \ldots, K$ και $z = 0, \ldots, M\}$. Αν και δεν υπάρχει προφανές όριο για τις τιμές των x και y, μπορεί εύκολα να αποδειχθεί ότι για να παραμείνει το σύστημα κερδοφόρο το συνολικό απόθεμα δεν πρέπει να υπερβαίνει το $K = M\lambda_1 r/h$ και ως εκ τούτου $x + y \leq K$. Σε κάθε ολοκλήρωση της παραγωγής ενός προϊόντος λαμβάνουμε μια απόφαση παραγωγής $\pi_\mu(x, y, z)$, όπου $\pi_\mu(x, y, z) = 1$ αν συνεχίζουμε την παραγωγή στην κατάσταση (x, y, z) και $\pi_\mu(x, y, z) = 0$ διαφορετικά. Όταν ένα αντικείμενο απορρίπτεται, η κατάσταση του συστήματος δεν αλλάζει. Όταν φτάνει μια παραγγελία από πελάτη κατηγορίας 1, λαμβάνουμε μια απόφαση πώλησης για τακτικούς πελάτες $\pi_{\lambda 1}(x, y, z)$, όπου $\pi_{\lambda 1}(x, y, z) = 1$ αν αποφασίσουμε να καλύψουμε την παραγγελία με προϊόν ποιότητας επιπέδου 1 ή $\pi_{\lambda 1}(x, y, z) = 2$ αν αποφασίσουμε να πουλήσουμε προϊόν ποιότητας επιπέδου 2. Αν ο πελάτης είναι ικανοποιημένος με την ποιότητα του αντικειμένου, τότε παραμένει τακτικός και η κατάσταση του συστήματος αλλάζει σε $(x - 1, y, z)$ ή $(x, y - 1, z)$ αντίστοιχα. Σε περίπτωση δυσαρέσκειας, ο πελάτης γίνεται περιστασιακός και η κατάσταση του συστήματος αλλάζει σε $(x - 1, y, z - 1)$ ή $(x, y - 1, z - 1)$ αντίστοιχα. Όταν $x = 0$ έχουμε μια επιπλέον επιλογή να μην εξυπηρετήσουμε την παραγγελία και συνεπώς να δυσαρεστήσουμε τον πελάτη, αλλάζοντας έτσι την κατάσταση του συστήματος σε $(x, y, z - 1)$. Όταν έχουμε παραγγελία από πελάτες κατηγορίας 2, λαμβάνουμε απόφαση πώλησης για περιστασιακούς πελάτες $\pi_{\lambda 2}(x, y, z)$, όπου $\pi_{\lambda 2}(x, y, z) = 1$ αν αποφασίσουμε να καλύψουμε την παραγγελία με προϊόν ποιότητας επιπέδου 1, ή $\pi_{\lambda 2}(x, y, z) = 2$ αν πουληθεί προϊόν ποιότητας επιπέδου 2, ή τελικά $\pi_{\lambda 2}(x, y, z) = 0$ αν αποφασίσουμε να απορρίψουμε την εισερχόμενη παραγγελία. Ένας πελάτης που είναι ικανοποιημένος με την ποιότητα του αντικειμένου γίνεται τακτικός πελάτης και η κατάσταση του συστήματος αλλάζει σε $(x - 1, y, z + 1)$ ή $(x, y - 1, z + 1)$ αντίστοιχα. Σε περίπτωση δυσαρέσκειας ο πελάτης παραμένει περιστασιακός και η κατάσταση του συστήματος αλλάζει σε $(x - 1, y, z)$ ή $(x, y - 1, z)$ αντίστοιχα. Σε περίπτωση απόρριψης της παραγγελίας, η κατάσταση του συστήματος δεν αλλάζει.

4.2.1 Δυναμικός προγραμματισμός

Η εξέλιξη του συστήματος καθορίζεται από δύο τύπους γεγονότων: την τοποθέτηση παραγγελίας από έναν πελάτη τύπου k και την παραγωγή ενός προϊόντος ποιότητας i. Χρησιμοποιώντας την τεχνική ομοιομορφοποίησης (βλ. [30], [35], [3] [37]) διατυπώνουμε το πρόβλημα ως μια διαδικασία λήψης αποφάσεων Markov. Το σταθερό μέγεθος M της αγοράς, καθώς και το συνολικό όριο αποθεμάτων K, περιορίζουν τον χώρο καταστάσεων και διασφαλίζουν τη σταθερότητα του συστήματος. Η συνεχής αλυσίδα Markov ομογενοποιείται χρησιμοποιώντας τον μέγιστο ρυθμό μετάβασης $v = M\lambda_1 + \mu$, που είναι ένας ομοιόμορφος ρυθμός εμφάνισης γεγονότων και επιτρέπει τη μετατροπή του συνεχούς χρόνου σε ακολουθία σταδίων. Ακολουθώντας την τυπική προσέγγιση του στοχαστικού δυναμικού προγραμματισμού, ορίζουμε το $V_k(x, y, z)$ ως το συνολικό αναμενόμενο κέρδος κατά τα πρώτα k γεγονότα (ή στάδια λήψης αποφάσεων) όταν η αρχική κατάσταση είναι (x, y, z). Οι εξισώσεις Bellman για το βέλτιστο συνολικό αναμενόμενο κέρδος κατά τα πρώτα $k + 1$ στάδια είναι:

$$\begin{aligned} V_{k+1}(0,0,0) = \frac{1}{v}\{&\mu \max[p_3(V_k(0,0,0) - c) + p_1 V_k(1,0,0) + p_2 V_k(0,1,0), V_k(0,0,0)] \\ &+ M\lambda_1 V_k(0,0,0)\} \end{aligned} \tag{4.1}$$

$$\begin{aligned} V_{k+1}(0,0,z) = \frac{1}{v}\{&\mu \max[p_3(V_k(0,0,z) - c) + p_1 V_k(1,0,z) + p_2 V_k(0,1,z), V_k(0,0,z)] \\ &+ z\lambda_1 V_k(0,0,z-1) + (M - z)\lambda_1 V_k(0,0,z)\},\ 1 \le z \le M \end{aligned} \tag{4.2}$$

$$\begin{aligned} V_{k+1}(x,0,0) = \frac{1}{v}\{&\mu \max[p_3(V_k(x,0,0) - c) + p_1 V_k(x+1,0,0) + p_2 V_k(x,1,0), V_k(x,0,0)] \\ &+ M\lambda_2 \max[s_1^2 V_k(x-1,0,1) + (1 - s_1^2) V_k(x-1,0,0) + r, V_k(x,0,0)] \\ &+ M(\lambda_1 - \lambda_2) V_k(x,0,0) - xh\},\ 0 < x < K \end{aligned} \tag{4.3}$$

$$\begin{aligned} V_{k+1}(0,y,0) = \frac{1}{v}\cdot \\ \{&\mu \max[p_3(V_k(0,y,0) - c) + p_1 V_k(1,y,0) + p_2 V_k(0,y+1,0), V_k(0,y,0)] \\ &+ M\lambda_2 \max[s_2^2 V_k(0,y-1,1) + (1 - s_2^2) V_k(0,y-1,0) + r, V_k(0,y,0)] \\ &+ M(\lambda_1 - \lambda_2) V_k(0,y,0) - yh\}, 0 < y < K \end{aligned} \tag{4.4}$$

$$\begin{aligned} V_{k+1}(x,y,0) = \frac{1}{v}\cdot \\ \{&\mu \max[p_3(V_k(x,y,0) - c) + p_1 V_k(x+1,y,0) + p_2 V_k(x,y+1,0), V_k(x,y,0)] \\ &+ M\lambda_2 \max[s_1^2 V_k(x-1,y,1) + (1 - s_1^2) V_k(x-1,y,0) + r, \\ &\qquad s_2^2 V_k(x,y-1,1) + (1 - s_2^2) V_k(x,y-1,0) + r, V_k(x,y,0)] \\ &+ M(\lambda_1 - \lambda_2) V_k(x,y,0) - (x+y)h\}, x > 0, y > 0, x + y < K \end{aligned} \tag{4.5}$$

$$
\begin{aligned}
V_{k+1}(0,y,z) = {} & \frac{1}{v}\cdot \\
& \{\mu \max[p_3(V_k(0,y,z)-c)+p_1V_k(1,y,z)+p_2V_k(0,y+1,z),V_k(0,y,z)] \\
& +z\lambda_1 \max[s_2^1V_k(0,y-1,z)+(1-s_2^1)V_k(0,y-1,z-1)+r,V_k(0,y,z-1)] \\
& +(M-z)\lambda_2 \max[s_2^2V_k(0,y-1,z+1)+(1-s_2^2)V_k(0,y-1,z)+r,V_k(0,y,z)] \\
& +(M-z)(\lambda_1-\lambda_2)V_k(0,y,z)-yh\},0<y<K,0<z<M
\end{aligned} \tag{4.6}
$$

$$
\begin{aligned}
V_{k+1}(0,y,M) = {} & \frac{1}{v}\cdot \\
& \{\mu \max[p_3(V_k(0,y,M)-c)+p_1V_k(1,y,M)+p_2V_k(0,y+1,M),V_k(0,y,M)] \\
& +M\lambda_1 \max[s_2^1V_k(0,y-1,M)+(1-s_2^1)V_k(0,y-1,M-1)+r,V_k(0,y,M-1)] \\
& -yh\},0<y<K
\end{aligned} \tag{4.7}
$$

$$
\begin{aligned}
V_{k+1}(x,0,M) = {} & \frac{1}{v}\cdot \\
& \{\mu \max[p_3(V_k(x,0,M)-c)+p_1V_k(x+1,0,M)+p_2V_k(x,1,M),V_k(x,0,M)] \\
& +M\lambda_1[s_1^1V_k(x-1,0,M)+(1-s_1^1)V_k(x-1,0,M-1)+r]-xh\},\ 0<x<K
\end{aligned} \tag{4.8}
$$

$$
\begin{aligned}
V_{k+1}(x,0,z) = {} & \frac{1}{v}\{\mu \max[p_3(V_k(x,0,z)-c)+p_1V_k(x+1,0,z)+p_2V_k(x,1,z),V_k(x,0,z)] \\
& +z\lambda_1[s_1^1V_k(x-1,0,z)+(1-s_1^1)V_k(x-1,0,z-1)+r] \\
& +(M-z)\lambda_2 \max[s_1^2V_k(x-1,0,z+1)+(1-s_1^2)V_k(x-1,0,z)+r,V_k(x,0,z)] \\
& +(M-z)(\lambda_1-\lambda_2)V_k(x,0,z)-xh\},0<x<K,0<z<M
\end{aligned} \tag{4.9}
$$

$$
\begin{aligned}
V_{k+1}(x,y,M) = {} & \frac{1}{v}\{\mu \max[p_3(V_k(x,y,M)-c)+p_1V_k(x+1,y,M)+p_2V_k(x,y+1,M),V_k(x,y,M)] \\
& +M\lambda_1 \max[s_1^1V_k(x-1,y,M)+(1-s_1^1)V_k(x-1,y,M-1)+r, \\
& \qquad s_2^1V_k(x,y-1,M)+(1-s_2^1)V_k(x,y-1,M-1)+r] \\
& -(x+y)h\},x>0,y>0,x+y<K
\end{aligned} \tag{4.10}
$$

$$
\begin{aligned}
V_{k+1}(K,0,0) = {} & \frac{1}{v}\{M\lambda_2 \max[s_1^2V_k(K-1,0,1)+(1-s_1^2)V_k(K-1,0,0)+r,V_k(K,0,0)] \\
& +[\mu+M(\lambda_1-\lambda_2)]V_k(K,0,0)-Kh\}
\end{aligned} \tag{4.11}
$$

$$
\begin{aligned}
V_{k+1}(0,K,0) = {} & \frac{1}{v}\{M\lambda_2 \max[s_2^2V_k(0,K-1,1)+(1-s_2^2)V_k(0,K-1,0)+r,V_k(0,K,0)] \\
& +[\mu+M(\lambda_1-\lambda_2)]V_k(0,K,0)-Kh\}
\end{aligned} \tag{4.12}
$$

$$
\begin{aligned}
V_{k+1}(K,0,M) = {} & \frac{1}{v}\{M\lambda_1[s_1^1V_k(K-1,0,M)+(1-s_1^1)V_k(K-1,0,M-1)+r] \\
& +\mu V_k(K,0,M)-Kh\}
\end{aligned} \tag*{4.(13)}
$$

$$
\begin{aligned}
V_{k+1}(0,K,M) = {} & \frac{1}{v}\{M\lambda_1 \max[s_2^1V_k(0,K-1,M)+(1-s_2^1)V_k(0,K-1,M-1)+r, \\
& V_k(0,K,M-1)]+\mu V_k(0,K,M)-Kh\}
\end{aligned} \tag{4.14}
$$

$$
\begin{aligned}
V_{k+1}(0,K,z) = \frac{1}{v}\cdot & \\
&\{z\lambda_1 \max[s_2^1 V_k(0,K-1,z) + (1-s_2^1)V_k(0,K-1,z-1) + r, V_k(0,K,z-1)] \\
&+ (M-z)\lambda_2 \max[s_2^2 V_k(0,K-1,z+1) + (1-s_2^2)V_k(0,K-1,z) + r, V_k(0,K,z)] \\
&+ [\mu + (M-z)(\lambda_1 - \lambda_2)]V_k(0,K,z) - Kh\}, 0 < z < M
\end{aligned} \tag{4.15}
$$

$$
\begin{aligned}
V_{k+1}(K,0,z) = \frac{1}{v}&\{z\lambda_1[s_1^1 V_k(K-1,0,z) + (1-s_1^1)V_k(K-1,0,z-1) + r] \\
&+ (M-z)\lambda_2 \max[s_1^2 V_k(K-1,0,z+1) + (1-s_1^2)V_k(K-1,0,z) + r, V_k(K,0,z)] \\
&+ [\mu + (M-z)(\lambda_1 - \lambda_2)]V_k(K,0,z) - Kh\}, 0 < z < M
\end{aligned} \tag{4.16}
$$

$$
\begin{aligned}
V_{k+1}(x,K-x,0) = \frac{1}{v}&\{M\lambda_2 \max[s_1^2 V_k(x-1,K-x,1) + (1-s_1^2)V_k(x-1,K-x,0) + r, \\
&\qquad s_2^2 V_k(x,K-x-1,1) + (1-s_2^2)V_k(x,K-x-1,0) + r, \\
&\qquad V_k(x,K-x,0)] \\
&+ [\mu + M(\lambda_1 - \lambda_2)]V_k(x,K-x,0) - Kh\}, 0 < x < K
\end{aligned} \tag{4.17}
$$

$$
\begin{aligned}
V_{k+1}(x,K-x,M) = \frac{1}{v}\cdot & \\
&\{M\lambda_1 \max[s_1^1 V_k(x-1,K-x,M) + (1-s_1^1)V_k(x-1,K-x,M-1) + r, \\
&\qquad s_2^1 V_k(x,K-x-1,M) + (1-s_2^1)V_k(x,K-x-1,M-1) + r] \\
&+ \mu V_k(x,K-x,M) - Kh\}, 0 < x, K
\end{aligned} \tag{4.18}
$$

$$
\begin{aligned}
V_{k+1}(x,K-x,z) = \frac{1}{v}\cdot & \\
&\{z\lambda_1 \max[s_1^1 V_k(x-1,K-x,z) + (1-s_1^1)V_k(x-1,K-x,z-1) + r, \\
&\qquad s_2^1 V_k(x,K-x-1,z) + (1-s_2^1)V_k(x,K-x-1,z-1) + r] \\
&+ (M-z)\lambda_2 \max[s_1^2 V_k(x-1,K-x,z+1) + (1-s_1^2)V_k(x-1,K-x,z) + r, \\
&\qquad s_2^2 V_k(x,K-x-1,z+1) + (1-s_2^2)V_k(x,K-x-1,z) + r, \\
&\qquad V_k(x,K-x,z)] \\
&+ [\mu + (M-z)(\lambda_1 - \lambda_2))V_k(x,K-x,z] - Kh\}, 0 < x < K, 0 < z < M
\end{aligned} \tag{4.19}
$$

$$
\begin{aligned}
V_{k+1}(x,y,z) = \frac{1}{v}\cdot & \\
&\{\mu \max[p_3(V_k(x,y,z) - c) + p_1 V_k(x+1,y,z) + p_2 V_k(x,y+1,z), V_k(x,y,z)] \\
&+ z\lambda_1 \max[s_1^1 V_k(x-1,y,z) + (1-s_1^1)V_k(x-1,y,z-1) + r, \\
&\qquad s_2^1 V_k(x,y-1,z) + (1-s_2^1)V_k(x,y-1,z-1) + r] \\
&+ (M-z)\lambda_2 \max[s_1^2 V_k(x-1,y,z+1) + (1-s_1^2)V_k(x-1,y,z) + r, \\
&\qquad s_2^2 V_k(x,y-1,z+1) + (1-s_2^2)V_k(x,y-1,z) + r, V_k(x,y,z)] \\
&+ (M-z)(\lambda_1 - \lambda_2)V_k(x,y,z) - (x+y)h\}, 0 < x < K, 0 < y < K, 0 < z < M
\end{aligned} \tag{4.20}
$$

για $k = 1, 2, 3, \ldots$ και αρχικές τιμές $V_0(x, y, z) \equiv 0$ για κάθε $(x, y, z) \in S$.

Η εξίσωση 4.20 είναι η γενικότερη εξίσωση καθώς αναφέρεται στις εσωτερικές καταστάσεις, ενώ οι υπόλοιπες εξισώσεις (εξισώσεις 4.1 – 4.19) αναφέρονται στις συνοριακές καταστάσεις. Οι όροι μέσα στις αγκύλες στην Εξίσωση 4.20 αντιστοιχούν στην παραγωγή ενός τελικού προϊόντος (πρώτος όρος), όπου πρέπει να αποφασίσουμε αν θα συνεχίσουμε να παραγουμε, στην άφιξη ενός τακτικού ή περιστασιακού πελάτη (δεύτερος και τρίτος όρος αντίστοιχα), όπου πρέπει να αποφασίσουμε τι είδους προϊόν πρέπει να χρησιμοποιήσουμε για να ικανοποιήσουμε την παραγγελία εφόσον αποφασίσουμε να την ικανοποιήσουμε (στην περίπτωση των περιστασιακών πελατών), σε μια αυτομετάβαση (ψευδογεγονός, τέταρτος όρος) και, τέλος, στο κόστος αποθεματοποίησης (πέμπτος όρος).

Η βέλτιστη μακροχρόνια μέση συνάρτηση κέρδους μπορεί να προσεγγιστεί αριθμητικά από τον αλγόριθμο διαδοχικών προσεγγίσεων, όπως αναφέρεται στην ακόλουθη πρόταση στην Πρόταση 2.1 του Κεφαλαίου 2 (Puterman [35]). Όπως έχουμε δει βασική προϋπόθεση για την Πρόταση 2.1 είναι ότι κάθε βέλτιστη αναμενόμενη στάσιμη ντετερμινιστική πολιτική έχει έναν απεριοδικό πίνακα μεταβάσεων. Όπως και στα προηγούμενα συστήματα που εξετάσαμε, ο πίνακας μεταβάσεων κάθε πολιτικής είναι απεριοδικός επειδή η πιθανότητα να παραμείνει το σύστημα στην τρέχουσα κατάσταση είναι θετική ανεξάρτητα από την απόφαση που λαμβάνεται.

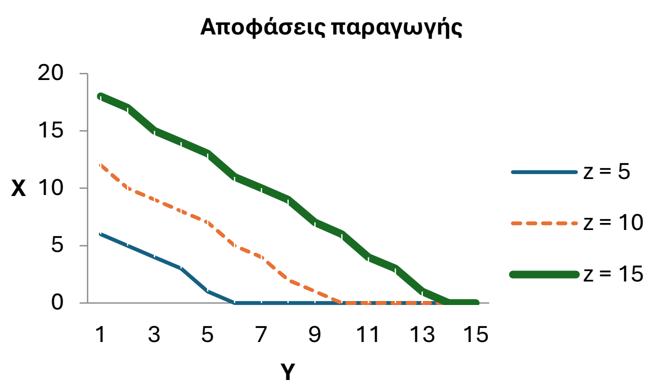
4.3 Αριθμητικά αποτελέσματα

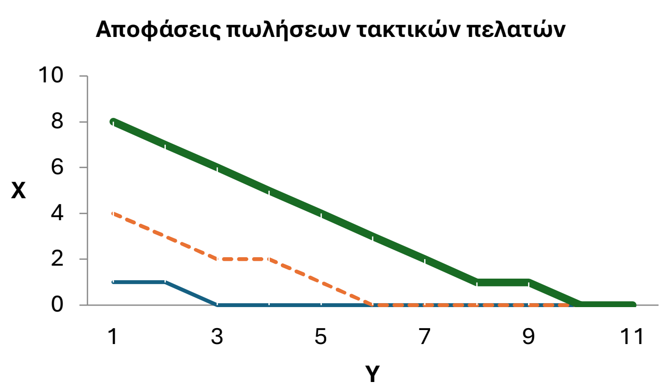
Για να διερευνήσουμε τη δομή της βέλτιστης πολιτικής, εκτελέσαμε μια σειρά από αριθμητικά πειράματα εφαρμόζοντας τον αλγόριθμο διαδοχικών προσεγγίσεων στις εξισώσεις 4.1 – 4.20. Για τις αριθμητικές δοκιμές χρησιμοποιήσαμε τις ακόλουθες τιμές παραμέτρων: $M = 15$, $\lambda_1 = 1$, $\lambda_2 = 0.1$, $\mu = 6$, $r = 1$, $c = 2$ και $h = 0.2$. Οι πιθανότητες ποιότητας προϊόντων και ικανοποίησης πελατών που χρησιμοποιήθηκαν στα αριθμητικά πειράματα παρουσιάζονται στον Πίνακα 4.1.

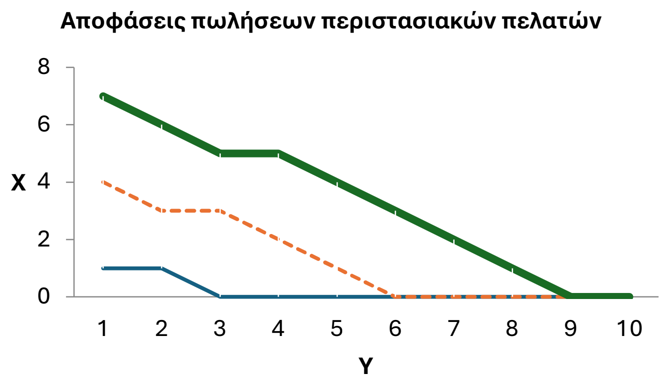
Πίνακας 4.1. Πιθανότητες παραγωγής και ικανοποίησης πελατών για το εξεταζόμενο παράδειγμα.

Επίπεδο ποιότητας *i*:	1	2
p_i	0.682689	0.271810
s_i^1	0.619317	0.195921
s_i^2	0.577068	0.145328

Στο Σχήμα 4.1 παρουσιάζουμε τις βέλτιστες πολιτικές για αυτό το σύστημα όταν ο αριθμός των τακτικών πελατών, z, είναι 5, 10 ή 15. Λόγω της πολυπλοκότητας του χώρου καταστάσεων και για να έχουμε καλύτερη παρουσίαση, επιλέξαμε να παρουσιάσουμε τις βέλτιστες πολιτικές για ορισμένες χαρακτηριστικές τιμές του z. Οι καμπύλες εναλλαγής αποφάσεων είναι τα όρια που χωρίζουν τις βέλτιστες αποφάσεις παραγωγής 1 (συνέχιση) και 0 (σταμάτημα παραγωγής) και τις βέλτιστες αποφάσεις πώλησης για προϊόντα ποιότητας 1 ή ποιότητας 2. Οι αποφάσεις παραγωγής 1 είναι βέλτιστες για μικρές τιμές του x και του y και επομένως επιλέγονται για σημεία (x, y) στα κάτω αριστερά ημιεπίπεδα του πρώτου διαγράμματος του Σχήματος 1. Τόσο για τους τακτικούς όσο και για τους περιστασιακούς πελάτες, είναι βέλτιστο να πωλούνται προϊόντα υψηλής ποιότητας όταν υπάρχει αρκετό απόθεμα και προϊόντα μεσαίας ποιότητας όταν το επίπεδο αποθέματος είναι χαμηλό. Επομένως, οι $\pi_{\lambda 1}(x, y, z) = 1$ και $\pi_{\lambda 2}(x, y, z) = 1$ είναι βέλτιστες για σημεία (x, y) στα πάνω δεξιά ημιεπίπεδα που ορίζονται από τα όρια που φαίνονται στα τελευταία δύο διαγράμματα του Σχήματος 4.1. Η απόρριψη ενός περιστασιακού πελάτη, δηλαδή η απόφαση $\pi_{\lambda 2}(x, y, z) = 0$, δεν είναι ποτέ βέλτιστη για τις τιμές των παραμέτρων που δίνονται παραπάνω. Φυσικά, στην κατάσταση $x = y = 0$ όπου δεν υπάρχει καθόλου απόθεμα, η παραγγελία του πελάτη απορρίπτεται από προεπιλογή.



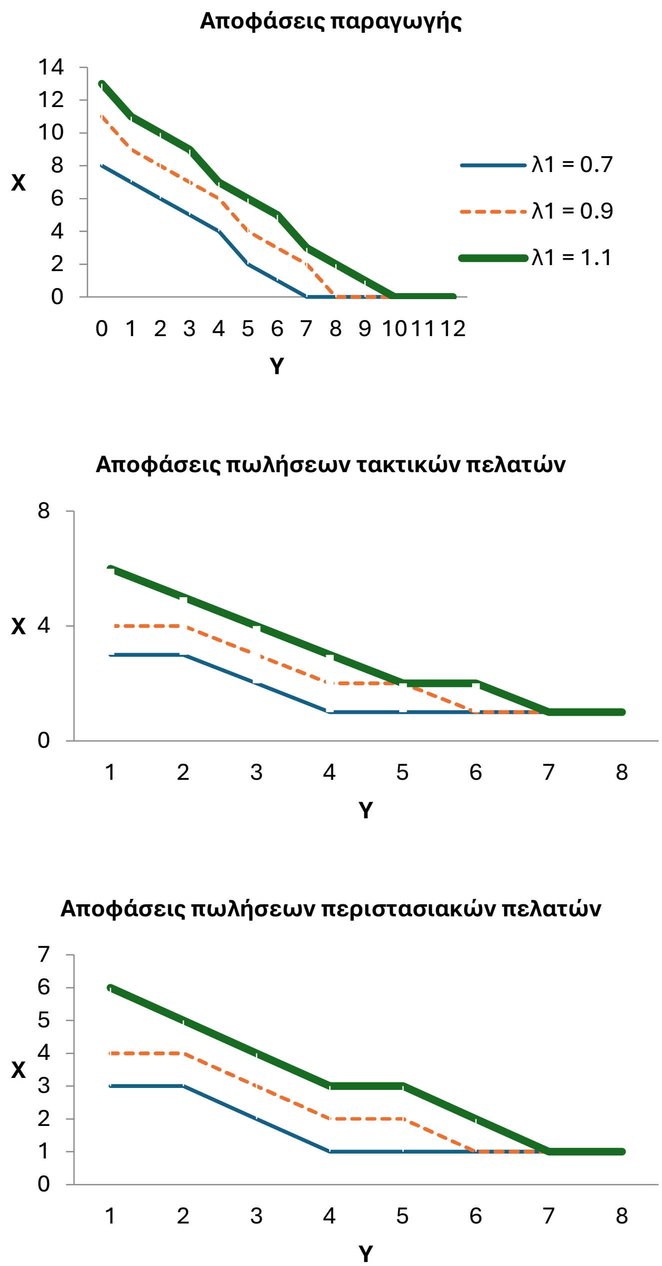




Σχήμα 4.1 Βέλτιστες πολιτικές για διαφορετικά επίπεδα τακτικών πελατών.

Αυτά τα αποτελέσματα είναι αναμενόμενα. Καθώς αυξάνεται ο αριθμός των τακτικών πελατών, αυξάνεται και ο συνολικός ρυθμός ζήτησης και, ως εκ τούτου, είναι απαραίτητο ένα υψηλότερο απόθεμα για να ικανοποιηθούν οι εισερχόμενες παραγγελίες. Επιπλέον, σε τέτοιες περιπτώσεις, μπορεί να είναι πιο κερδοφόρο να πωλούνται προϊόντα χαμηλότερης ποιότητας ($i = 2$) πιο συχνά από ό,τι όταν ο ρυθμός ζήτησης είναι χαμηλός. Αξίζει να σημειωθεί ότι ο αλγόριθμος διαδοχικών προσεγγίσεων μας δίδει $J^* = 1.6891$ ως τον βέλτιστο μέσο ρυθμό κέρδους για το αρχικό παράδειγμα.

Στη συνέχεια, εξετάζουμε πόσο ευαίσθητες είναι οι βέλτιστες πολιτικές στις μεταβολές του ρυθμού ζήτησης λ_1 των τακτικών πελατών. Στο Σχήμα 4.2 παρουσιάζουμε τις βέλτιστες αποφάσεις όταν ο αριθμός των τακτικών πελατών είναι $z = 10$, και για τρεις διαφορετικές τιμές του λ_1. Τα αποτελέσματα είναι και πάλι αρκετά λογικά, καθώς ένας υψηλότερος ρυθμός ζήτησης απαιτεί υψηλότερα επίπεδα αποθεμάτων και, δεδομένου ότι οι πιθανότητες ποιότητας εξερχόμενων προϊόντων και ο ρυθμός παραγωγής του συστήματος είναι σταθεροί, αυτό είναι δυνατό μόνο με την αύξηση της ποσότητας των αντικειμένων στο απόθεμα, άρα αποφασίζοντας $\pi(x, y, z) = 1$ πιο συχνά, και χαλαρώνοντας τις απαιτήσεις ποιότητας και λαμβάνοντας αποφάσεις $\pi_{\lambda 1}(x, y, z) = 2$ και $\pi_{\lambda 2}(x, y, z) = 2$ επίσης συχνότερα.



Σχήμα 4.2 Βέλτιστες πολιτικές για διάφορα επίπεδα ρυθμού ζήτησης τακτικών πελατών λ_1 όταν $z = 10$.

4.4 Σύνοψη

Αναπτύξαμε ένα μοντέλο απλού συστήματος παραγωγής προς αποθεματοποίηση το οποίο ενσωματώνει την ικανοποίηση των πελατών και τη δυναμική της αγοράς στις αποφάσεις παραγωγής και ελέγχου ποιότητας. Το μοντέλο μπορεί να χρησιμεύσει ως εργαλείο για τη διερεύνηση βέλτιστων πολιτικών για συστήματα παραγωγής. Επιπλέον, υπογραμμίζει τις αλληλεπιδράσεις μεταξύ παραγωγής, πωλήσεων και ποιότητας, δείχνοντας τις επιπτώσεις της ποιότητας στην ικανοποίηση των πελατών και επομένως στην κερδοφορία του συστήματος

Το σύστημα που παρουσιάστηκε σε αυτό το κεφάλαιο είναι ιδιαίτερα σύνθετο και δεν φαίνεται να υπάρχουν σχετικά εύκολες επεκτάσεις. Παρότι φαίνεται να υπάρχουν σχετικά απλές ευρετικές πολιτικές, που θα μπορούσε να προτείνει κανείς ως εναλλακτικές της βέλτιστης πολιτικής, δεν φαίνεται να υπάρχουν κατάλληλα αναλυτικά εργαλεία για την αποτελεσματική μαθηματική περιγραφή και ανάλυση τους. Επίσης παρουσιάζεται ιδιαίτερα δύσκολη η επέκταση αυτής της προσέγγισης σε σύνθετα συστήματα πολλών σταδίων. Η χρήση προσεγγιστικών τεχνικών όπως η προσομοίωση και η ενισχυτική μάθηση παρουσιάζεται ως μονόδρομος.

5 Συμπεράσματα

Σε αυτή την διατριβή αναπτύξαμε μαθηματικά μοντέλα για να κατανοήσουμε πώς οι έλεγχοι ποιότητας και παραγωγής μπορεί να επηρεάσουν την ικανοποίηση των πελατών και την κερδοφορία σε συστήματα παραγωγής. Εξετάσαμε τρία προβλήματα:

- έλεγχος **παραγωγής** και **ποιότητας** σε δίκτυα παραγωγής κατά παραγγελίες (make-to-order),
- έλεγχος **παραγωγής** και **ποιότητας** σε συστήματα ενός σταδίου που παράγουν κατά παραγγελίες (make-to-order), και οι πελάτες έχουν μνήμη της κατάστασης τους πριν αγοράσουν ένα προϊόν,
- έλεγχος **παραγωγής** και **ποιότητας** σε συστήματα ενός σταδίου που παράγουν προς αποθεματοποίηση (make-to-stock).

Και στα τρία προβλήματα χρησιμοποιήσαμε εργαλεία στοχαστικού δυναμικού προγραμματισμού για να διερευνήσουμε αριθμητικά την δομή της βέλτιστης πολιτικής. Σε όλα τα υπό εξέταση προβλήματα η δομή της βέλτιστης πολιτικής αποδείχτηκε αρκετά περίπλοκη. Παρόλα αυτά προέκυψαν χρήσιμα συμπεράσματα, που είναι δυνατόν να οδηγήσουν σε σχετικά απλές αλλά αποτελεσματικές ευρετικές πολιτικές. Αυτή είναι η περίπτωση των δύο πρώτων προβλημάτων, που εξετάζουμε συστήματα παραγωγής κατά παραγγελία, όπου προτείναμε απλές ευρετικές πολιτικές τύπου κατωφλίου. Οι προτεινόμενες πολιτικές μπορούν να αναλυθούν με την χρήση εργαλείων από την θεωρία ουρών αναμονής και πιο συγκεκριμένα από την θεωρία κλειστών δικτύων αναμονής. Τα εργαλεία αυτά μπορούν να χρησιμοποιηθούν σε περίπλοκα δίκτυα παραγωγής αφού έχουν μικρές υπολογιστικές απαιτήσεις ενώ οι προτεινόμενες πολιτικές προσεγγίζουν ικανοποιητικά την απόδοση των βέλτιστων πολιτικών.

Όπως αναφέραμε και στα προηγούμενα κεφάλαια η επιτυχία των προτεινόμενων πολιτικών μας ενθαρύνει να προτείνουμε αντίστοιχες ευρετικές πολιτικές και στα επόμενα προβλήματα που μελετώνται σε αυτή την διατριβή. Μια άλλη κατεύθυνση προς την οποία θα μπορούσε να κινηθεί μελλοντικά η έρευνα σε αυτό το πεδίο, είναι η περίπτωση που η ικανοποίηση πελατών εξαρτάται όχι μόνο από την ποιότητα των παραγόμενων προϊόντων αλλά και από τον χρόνο εξυπηρέτησης των εισερχομένων παραγγελιών. Η συνθετότητα των προβλημάτων που προκύπτουν σε αυτή την περίπτωση οδηγεί στην χρήση εργαλείων όπως η προσομοίωση και προσεγγιστικές τεχνικές όπως η ενισχυτική μάθηση. Οι προσεγγίσεις αυτές μπορούν να χρησιμοποιηθούν ακόμη και αν αδύνατο να χρησιμοποιήσουμε αναλυτικά ή ακριβή αριθμητικά εργαλεία, όπως οι αλυσίδες Markov και οι μαρκοβιανές διαδικασίες απόφασης.

Τα αποτελέσματα της διατριβής αυτής περιλαμβάνονται και στις εργασίες Konstantas et al. [21], [22], [23] και [24].

Βιβλιογραφία

[1] Anderson, E.W., and Sullivan, M. (1993) The antecedents and consequences of customer satisfaction for firms, *Marketing Science*, **12**, 125–143. doi: 10.1287/mksc.12.2.125

[2] Anderson, E.W., Fornell, C, Lehmann D.R. (1994) Customer satisfaction, market share and profitability. *Journal of Marketing*, **58**, 53–66. doi: 10.2307/1252310

[3] Bertsekas, D. (1995) *Dynamic Programming and Optimal Control – Vol. 2.* Athena Scientific, Belmont, MA, USA.

[4] Boucherie, R.J. and Van Dijk, N.M. (2017) *Markov decision processes in practice*, Springer. doi: 10.1007/978-3-319-47766-4

[5] Buzacott, J.A., and Shanthikumar, J.G. (1993) *Stochastic models of manufacturing systems* Prentice Hall, Englewood Cliffs, New Jersey.

[6] Buzen, J.P. (1973) Computational algorithms for closed queueing networks with exponential servers. *Communications of the ACM*, **16**, 527–531. doi: 10.1145/362342

[7] Colledani, M. Tolio, T. (2011) Integrated analysis of quality and production logistics per-formance in manufacturing lines, *International Journal Production Research*, **49**,485–518. doi: 10.1080/00207540903443246

[8] Deng, T., Shen, Z.-J.M., and Shanthikumar, J.G. (2014) Statistical learning of service-dependent demand in a multiperiod newsvendor setting, *Operations Research*, **62**, 1064–1076. doi: 10.1287/opre.2014.1303

[9] Fornell, C. and Wernerfelt, B. (1987) Defensive marketing strategy by customer complaint management: a theoritical analysis. *Journal of Marketing Research*, **24**, 337–346. doi: 10.1177/002224378702400401

[10] Fornell, C., and Wernerfelt, B. (1988) A model for customer complaint mangement. *Marketing Science*, **7**, 271–286. doi: 10.1287/mksc.7.3.287.

[11] Gershwin, S.B., Schick, I. (2007) A taxonomy of quality/quantity issues in manufacturing systems. *Proceedings of the 6th Conference on the Analysis of Manufacturing Sys-tems*, Lunteren, The Netherlands, 2007.

[12] Giannakis, G. Kouikoglou, V.S. and Nikitas, S. (2010) Workflow analysis of production lines with complete inspection and rework loops. *International Journal of Production Research*, **48**, 7321–7336. doi: 10.1080/00207540903469050

[13] Govil, M.K. and Fu, M.C. (1999) Queueing theory in manufacturing: a survey. *Journal of Manufacturing Systems*, **18**, 214–240. doi: 10.1016/S0278-6125(99)80033-8

[14] Grigoroudis, E., and Siskos, Y. (2010) *Customer satisfaction evaluation: Methods for measuring and implementing service quality*, Springer, New York

[15] Hajji, A. Mhada, F. Gharbi, A. Pellerin, R. Malhame, R. (2011) Integrated product specifications and productivity decision making in unreliable manufacturing systems. *International Journal of Production Economics*, **129**, 32–42. doi: 10.1016/j.ijpe.2010.08.008

[16] Hajji, A, Gharbi, A. and Pellerin, A. (2012) Joint production control and product quality deci-sion making in a failure prone multiple-product manufacturing system. *International Journal of Production Research*, **50**, 3661–3672. doi: 10.1080/00207543.2012.671588

[17] Inman, R.R. Blumenfeld, D.E. Huang, N. Li, J. (2013) Survey of recent advances on the interface between production system design and quality, *IIE Transactions*, **45**, 557–574. doi: 10.1080/0740817x.2012.757680

[18] Ioannidis, S. Kouikoglou, V.S. and Phillis, Y.A. (2004) Coordinating quality, production and sales in manufacturing systems, *International Journal of Production Research*, **42**, 3947–3956. doi: 10.1080/00207540410001696357

[19] Ittner, C.D., and Larcker, D.F. (1998) Are non-financial measures leading indicators of financial performance? An analysis of customer satisfaction. *Journal of Accounting Research*, **36**, 1–35. doi: 10.2307/2491304

[20] Kim, J., and Gershwin, S.B. (2005) Integrated quality and quantity modeling of a production line, *OR Spectrum*, **27**, 287–314. doi: 10.1007/s00291-005-0202-1

[21] Konstantas, D. Grigoroudis, E. Kouikoglou, V.S. and Ioannidis, S. (2015) A simple model of the effects of quality on market share and profitability in single stage manufacturing systems. *Proceedings of the 10th Conference on Stochastic Models of Manufacturing and Service Operations*, Volos, Greece, 2015.

[22] Konstantas, D. Ioannidis, S. Grigoroudis, E. and Kouikoglou, V.S. (2017). The Effects of Quality on Market Share and Profitability in Single Stage Make-to-Stock Production Systems. In *Operational Research in Business and Economics: 4th International Symposium and 26th National Conference on Operational Research,* Chania, Greece, June 2015 (pp. 235-245). Springer International Publishing. doi: 10.1007/978-3-319-33003-7_12

[23] Konstantas, D. Ioannidis, S. Kouikoglou, V.S. and Grigoroudis, E. (2018). Linking product quality and customer behavior for performance analysis and optimization of make-to-order manufacturing systems. *The International Journal of Advanced Manufacturing Technology*, **95**, 587–596. doi: 10.1007/s00170-017-1225-x

[24] Konstantas, D. Ioannidis, S. Grigoroudis, E. and Kouikoglou, V.S. (2018) A Study on the Effects of Quality on Market Share in Single Stage Manufacthuring Systems with Customer Class Dependent Satisfaction. *In 7th International Symposium and 29th National Conference on Operational Research The contribution of Operational Research, new technologies and innovation in agriculture and tourism.*

[25] Kostami, V. and Rajagopalan, S. (2014) Speed-quality trade-offs in a dynamic model. *Manufacturing and Service Operations Management*, **16**, 104–118. doi: 10.1287/msom.2013.0458

[26] Kouikoglou, V.S. and Phillis, Y.A. (2002) Design of product specifications and control policies in a single-stage production system, *IIE Transactions*, **34**, 590–600. doi: 10.1080/07408170208928896

[27] Li, J. Blumenfeld, D.E. Marin, S.P. (2008) Production system design for quality robustness, *IIE Transactions*, **40**, 162–176. doi: 10.1080/07408170601013661

[28] Liberopoulos., G., Tsikis, I. and Kozanidis, G. (2005) Inventory control in a duopoly: A dynamic non-cooperative game-theoretic approach. *Proceedings of the 17th National Conference of the Hellenic Operational Research Society (HELORS 2005)*, Patras, Greece, June (pp. 16–18). https://www.researchgate.net/profile/Isidoros-Tsikis/publication/228988398_Inventory_Control_in_a_Duopoly_A_Dynamic_Non-Cooperative_Game-Theoretic_Approach/links/0912f5080f0c2643da000000/Inventory-Control-in-a-Duopoly-A-Dynamic-Non-Cooperative-Game-Theoretic-Approach.pdf

[29] Liberopoulos, G. and Deligiannis, M. (2022). Optimal supplier inventory control policies when buyer purchase incidence is driven by past service. *European Journal of Operational Research*, **300**, 917–936. doi: 10.1016/j.ejor.2021.09.002

[30] Lippman, S.A. (1975) Applying a new device in the optimization of exponential queuing systems. *Operations Research*, **23**, 687–710. doi: 10.1287/opre.23.4.687

[31] Morse, P. (1958) *Queues, inventories and maintenance* Wiley, New York.

[32] Oliver, R. (1997) *Satisfaction: A Behavioral Perspective on the Customer* McGraw-Hill, New York. doi: 10.4324/9781315700892

[33] Papadopoulos, C.T. Li, J. and O'Kelly, M.E. (2019) A classification and review of timed Markov models of manufacturing systems. *Computers and Industrial Engineering*, **128**, 219-244. doi: 10.1016/j.cie.2018.12.019

[34] Paraschos, P.D. Koulinas, G.K. and Koulouriotis, D.E. (2020). Reinforcement learning for combined production-maintenance and quality control of a manufacturing system with deterioration failures, *Journal of Manufacturing Systems*, **56**, 470–483. doi: 10.1016/j.jmsy.2020.07

[35] Puterman, M.L. (1994) *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, New York. doi: 10.1002/9780470316887

[36] Rust, R.T. and Zahorik, A.J. (1993) Customer satisfaction, customer retention, and market share, *Journal of Retailing*, **57**, 25–48. doi: 10.1016/0022-4359(93)90003-2

[37] Sennott, L.I. (1999) *Stochastic Dynamic Programming and the Control of Queueing Systems*. Wiley, New York, NY, USA.

[38] Shin, S. Kongsuwon, P. and Cho, B.R. (2010) Development of the parametric tolerance modeling and optimization schemes and cost-effective solutions, *European Journal of Operational Research*, **207**, 1728–1741. doi:10.1016/j.ejor.2010.07.009

[39] Taguchi, G. Elsayed, E. and Hsiang, T. (1989) *Quality Engineering in Production Systems* McGraw-Hill, New York.

[40] Tan, B. Karabağ, O. and Khayyati, S. (2024). Energy-efficient production control of a make-to-stock system with buffer-and time-based policies. *International Journal of Production Research*, **62**, 5809–5827. doi: 10.1080/00207543.2023.2298488

[41] Tang, K. and Tang, J. (1994) Design of screening procedures: a review, *Journal of Quality Technology*, **26**, 209–226. doi: 10.1080/00224065.1994.11979527

[42] Van Ryzin, G. Lou, S.X.C. and Gershwin, S.B. (1993) Production control for a tandem two-machine system. *IIE Transactions*, **25**, 5–20.

[43] Veach, M.H. and Wein, L.M. (1992) Monotone control of queueing networks. *Queueing Systems*, **12**, 391–408.

[44] Veach, M.H. and Wein, L.M. (1994) Optimal control of a two-station tandem production-inventory system. *Operations Research*, **42**, 337–350. https://www.jstor.org/stable/171676

[45] Weber, R.R. and Stidham, S. Jr (1987) Optimal control of service rates in networks of queues. *Advances in Applied Probability*, **19**, 202–218.

[46] Yao, D.D., and Zheng, S. (2002) *Dynamic control of quality in production-inventory systems: coordination and optimization* Springer – Verlag, New York. doi: 10.1007/b98974

Παράρτημα A1

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης πολιτικής στο πρόβλημα ενός σταδίου της παραγράφου 2.4.1. Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos
Public x As Double              'taktikoi pelates
Public y As Double              'pelates sto systima
'M-x-y : oi efimeroi pelates ektos systymatos
Public lev As Integer           'epipeda poititas
Public p() As Double           'dianisma pithanotitwn epipedwn poiotitas
Public s() As Double                'dianisma pithanotitwn ikanopoiisis
pelatwn apo agora proiontwn apo ta epipeda poiotitas
Public lamda(2) As Double        'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi As Double             'rythmos eksypiretisis
Public c As Double              'monadiaio kostos aporripsis kommatiou
Public r As Double              'monadiao ofelos pwlisis proiontos
Public b As Double                'monadiaio kostos anamonis ana monada
xronou
Public vmax As Double           'megistos synolikos rythmos afiksis
Public vold() As Double        'pinakas me ta esoda gia kathe katastasi
ti xroniki stigmi k
Public vnew() As Double     'pinakas me ta esoda gia kathe katastasi ti
xroniki stigmi k+1
Public dis() As Integer       'pinakas me ti metavliti apofasis gia tis
antistoixes katastaseis: 1=poulaw, 0=petaw
Public error As Double         'apoklisi kerdwn metaksy twn stigmwn k+1
kai k
Public merror As Double             'orio apoklisis gia to peras twn
epanalipsewn
Public rep As Long            'metritis epanalispewn
Public pen As Double            'penalti, kostos, kakis poiotitas
Public optavereven As Double    'meso veltisto kerdos
Public sign As Integer          'deiktis yparksis arnitikou kerdous

Sub main()

Dim i As Integer
Dim dif As Double           'Boithitikes metavlites
Dim maxdif As Double
Dim mindif As Double
Dim ath As Double          'metavliti gia to athroisma stous typous twn
V

'apodosi arxikwn timwn kai dedomenwn

Open App.Path & "\data.txt" For Input As #1           'Anoigma arxeiou
dedomenwn gia diavasma
 Input #1, M
```

```
 Input #1, lev
 ReDim vold(M, M)                                     'Epanakoa8orismos
megethous pinakwn
ReDim vnew(M, M)
ReDim dis(M, M)
ReDim p(lev)
ReDim s(lev)
 For i = lev To 0 Step -1
 Input #1, p(i)
 Next i
 For i = lev To 0 Step -1
 Input #1, s(i)
 Next i

Input #1, lamda(1)
Input #1, lamda(2)
Input #1, mi
Input #1, c
Input #1, r
Input #1, b
Input #1, pen
Input #1, merror

vmax = mi + M * lamda(1)

error = 1000000
rep = 0

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ results.txt" For Output As #filenum       'Anoigma
arxeiou apotelesmatwn

While (error < -merror) Or (error > merror)                 'Synthiki
epanalipsis
 rep = rep + 1
 For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1                          'Ypologismos
twn V(k+1) gia oles tis katastaseis
        If x = 0 Then
            If y = 0 Then
                'KATASTASI (0,0)
                vnew(0, 0) = (1 / vmax) * (M * lamda(2) * vold(0, 1) +
(M * (lamda(1) - lamda(2)) + mi) * vold(0, 0))
            ElseIf y = M Then
                ath = 0
                For i = 0 To lev Step 1     'Euresi athroismatos me max
sto typo
                    If (s(i) * vold(1, (M - 1)) + (1 - s(i)) * (vold(0,
(M - 1)) - pen) + r) >= (vold(0, M) - c) Then
                        ath = ath + mi * p(i) * (s(i) * vold(1, (M - 1))
+ (1 - s(i)) * (vold(0, (M - 1)) - pen) + r)
                    Else
                        ath = ath + mi * p(i) * (vold(0, M) - c)
                    End If
                Next i
                'KATASTASI (0,M)
                vnew(0, M) = (1 / vmax) * (ath + M * lamda(1) * vold(0,
M) - b * M)
            Else
```

```
                ath = 0
                For i = 0 To lev Step 1     'Euresi athroismatos me max
sto typo
                    If (s(i) * vold(1, y - 1) + (1 - s(i)) * (vold(0,
(y - 1)) - pen) + r) >= (vold(0, y) - c) Then
                        ath = ath + mi * p(i) * (s(i) * vold(1, (y - 1))
+ (1 - s(i)) * (vold(0, (y - 1)) - pen) + r)
                    Else
                        ath = ath + mi * p(i) * (vold(0, y) - c)
                    End If
                Next i
                'KATASTASI (0,y)
                vnew(0, y) = (1 / vmax) * ((M - y) * lamda(2) * vold(0,
(y + 1)) + ath + (M * (lamda(1) - lamda(2)) + y * lamda(2)) * vold(0,
y) - b * y)
            End If
        ElseIf y = 0 Then
            If x = M Then
                'KATASTASI (M,0)
                vnew(M, 0) = (1 / vmax) * (M * lamda(1) * vold((M - 1),
1) + mi * vold(M, 0))
            Else
                'KATASTASI (x,0)
                vnew(x, 0) = (1 / vmax) * (x * lamda(1) * vold((x - 1),
1) + (M - x) * lamda(2) * vold(x, 1) + ((M - x) * (lamda(1) - lamda(2))
+ mi) * vold(x, 0))
            End If
        ElseIf (x + y) = M Then
             ath = 0
             For i = 0 To lev Step 1        'Euresi athroismatos me max
sto typo
                 If (s(i) * vold((x + 1), (y - 1)) + (1 - s(i)) * (vold(x,
(y - 1)) - pen) + r) >= (vold(x, y) - c) Then
                        ath = ath + mi * p(i) * (s(i) * vold((x + 1),
(y - 1)) + (1 - s(i)) * (vold(x, (y - 1)) - pen) + r)
                 Else
                        ath = ath + mi * p(i) * (vold(x, y) - c)
                 End If
             Next i
             'KATASTASI (x,M-x)
             vnew(x, y) = (1 / vmax) * (x * lamda(1) * vold((x - 1), (y
+ 1)) + ath + (M - x) * lamda(1) * vold(x, y) - b * y)
        Else
             ath = 0
             For i = 0 To lev Step 1        'Euresi athroismatos me max
sto typo
                 If (s(i) * vold((x + 1), (y - 1)) + (1 - s(i)) * (vold(x,
(y - 1)) - pen) + r) >= (vold(x, y) - c) Then
                        ath = ath + mi * p(i) * (s(i) * vold((x + 1),
(y - 1)) + (1 - s(i)) * (vold(x, (y - 1)) - pen) + r)
                 Else
                        ath = ath + mi * p(i) * (vold(x, y) - c)
                 End If
             Next i
             'KATASTASI (x,y)
             vnew(x, y) = (1 / vmax) * (x * lamda(1) * vold((x - 1), (y
+ 1)) + (M - x - y) * lamda(2) * vold(x, (y + 1)) + ath + ((M - x) *
(lamda(1) - lamda(2)) + y * lamda(2)) * vold(x, y) - b * y)
        End If

      Next y
```

```
   Next x
   sign = 1
   maxdif = 0
   mindif = 1000000
   For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
        dif = vnew(x, y) - vold(x, y)   'Diafora sto V metaksy diadoxikwn
xronikwn stigmwn
        If (dif < 0) Then
            dif = -dif
            sign = -1
        Else
            sign = 1
        End If
        If (dif < mindif) Then
            mindif = dif
        End If
        If (dif > maxdif) Then
            maxdif = dif
        End If
    Next y
   Next x

   error = maxdif - mindif       'Kathorismos tis apoklisis


   optavereven = maxdif * vmax         'Ypologismos mesou kerdous

   For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
        vold(x, y) = vnew(x, y)        'Antikatastasi twn V metaksy twn
xronikwn stigmwn k kai k+1, gia metabasi se epomeni xroniki stigmi
    Next y
   Next x

Wend                                    'telos epanalipsis

Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print  #filenum,  "O  rythmos  afiksis  twn  eukairiakwn  pelatwn  einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis eimai mi=", mi
Print #filenum, "To kostos aporripsis enos kommatiou einai c=", c
Print #filenum, "To kostos anamonis enos pelati ana monada xronou einai
b=", b
Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai r=", r
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:",
lev + 1
Print #filenum, "H pithanotita paragwgis kai i pithanotita apodoxis gia
kathe epipedo poiotitas einai antistoixa:"
For i = 0 To lev Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s(i)
Next i
Print #filenum,
Print #filenum, "To orio sfalmatos oristike sto ", merror
Print #filenum, "Petyxame sygklisi meta apo arithmo epanalipsewn:", rep
Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum, "sign:", sign
```

```
Print #filenum,
For i = 0 To lev Step 1
Print #filenum, "O pinakas apofasewn tha einai gia to epipedo poiotitas
", i
    For x = (M - 1) To 0 Step -1
Print #filenum, x,
        For y = 1 To (M - x) Step 1                  'Epilogi apofasis
gia kathe katastasi
            If ((s(i) * vold((x + 1), (y - 1)) + (1 - s(i)) * (vold(x,
(y - 1)) - pen) + r) >= (vold(x, y) - c)) Then
                dis(x, y) = 1
            Else
                dis(x, y) = 0
            End If
            Print #filenum, dis(x, y),
        Next y
        Print #filenum, x
        Print #filenum,
    Next x
    Print #filenum, "y",
    For y = 1 To M Step 1
    Print #filenum, y,
    Next y
    Print #filenum, "y"
    Print #filenum,
Next i


End Sub
```

Παράρτημα A2

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης ευρετικής πολιτικής με την χρήση ΚΔΑ, στο πρόβλημα ενός σταδίου της παραγράφου 2.4.1 Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos
Public x As Double              'taktikoi pelates
Public y As Double              'pelates sto systima
'M-x-y : oi efimeroi pelates ektos systymatos
Public lev As Integer           'epipeda poititas
Public p() As Double           'dianisma pithanotitwn epipedwn poiotitas
Public s() As Double               'dianisma pithanotitwn ikanopoiisis
pelatwn apo agora proiontwn apo ta epipeda poiotitas
Public lamda(2) As Double         'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi(3) As Double             'rythmos eksypiretisis
Public c As Double              'monadiaio kostos aporripsis kommatiou
Public r As Double              'monadiao ofelos pwlisis proiontos
Public b As Double                'monadiaio kostos anamonis ana monada
xronou
Public bi() As Double              'ta bita gia tous komvous 2,3 (idia)
analoga me to plithos ston komvo
Public u(3) As Double           'ta u gia ka8e komvo
Public ath As Double              ' metritis gia to athroisma ston typo
tou G()
Public plithos As Integer       'to megethos N pou exoume mesa sto G()
Public xi(3) As Double          'ta x gia kathe komvo tou diktiou
Public ps() As Double           'pi8anotita na ikonopoii8ike ananola to
epipedo poiotitas pou exoume san katofli
Public pns() As Double         'pi8anotita na min ikanopoii8ika analoga
me to epipedo pou exoume san katofli
Public optavereven As Double    'meso veltisto kerdos
Public G1 As Double             'G(M-1)
Public G2 As Double             'G(M)
Public th1 As Double            'TH komvou1
Public meso As Double               'meso plithos pelatwn sto systima
(komvos1)
Public pk() As Double           'Pithanotites katastasewn
Public py() As Double               'pithanotites plithous pelatwn sto
systima
Public testaki As Double        'athroisma pithanotitwn katastasewn....
prepei na vgainei 1


Sub Main()

Dim i As Integer
Dim j As Integer
Dim z As Integer
Dim k As Integer
```

```
'apodosi arxikwn timwn kai dedomenwn
Open App.Path & "\data.txt" For Input As #1
 Input #1, M
 Input #1, lev

ReDim bi(M)
ReDim pk(M, M)
ReDim py(M)

x = 0
y = 0

'epanakathorismos megethos dianismatwn s kai p
ReDim p(lev)
ReDim s(lev)
ReDim ps(lev)
ReDim pns(lev)

'Dedomena gia tis pithanotites
For i = 7 To 0 Step -1
Input #1, p(i)
Next i

For i = lev To 0 Step -1
Input #1, s(i)
Next i

For i = 0 To lev Step 1
Input #1, ps(i)
Next i

For i = 0 To lev Step 1
Input #1, pns(i)
Next i

'Dedomena
Input #1, lamda(1)
Input #1, lamda(2)
Input #1, mi(1)
Input #1, c
Input #1, r
Input #1, b

'Ypologismos twn bita gia tous komvous 2,3
'Gia  ton  komvo  1,  ola  ta  bita  einai  =1  afou  exoume  mia  monada
eksypiretisis
bi(0) = 1
For i = 1 To M Step 1
    bi(i) = i * bi(i - 1)
Next i

mi(2) = lamda(1)
mi(3) = lamda(2)

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ fms_results.txt" For Output As #filenum      'Anoigma
arxeiou apotelesmatwn
```

```
Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print #filenum, "O rythmos afiksis twn eukairiakwn pelatwn einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis eimai mi=", mi(1)
Print #filenum, "To kostos aporripsis enos kommatiou einai c=", c
Print #filenum, "To kostos anamonis enos pelati ana monada xronou einai
b=", b
Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai r=", r
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:",
lev + 1
Print #filenum, "H pithanotita paragwgis kai i pithanotita apodoxis gia
kathe epipedo poiotitas einai antistoixa:"
For i = 0 To lev Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s(i)
Next i
Print #filenum,

'Ypologismos twn u gia kathe komvo
For k = 0 To 7 Step 1             'Gia ola ta pithana katoflia apodektou
epipedou poiotitas
    u(1) = 1 / (1 + ps(k) + pns(k))
    u(2) = ps(k) / (1 + ps(k) + pns(k))
    u(3) = pns(k) / (1 + ps(k) + pns(k))

    For i = 1 To 3
        xi(i) = u(i) / mi(i)
    Next i

    plithos = M
    Call Ypologismos_G
    G2 = ath
    plithos = M - 1
    Call Ypologismos_G
    G1 = ath

    th1 = u(1) * G1 / G2

 Print #filenum, "An theorisoume san katofli apodektis poiotitas to
epipedo", k
 Print #filenum, "To throughput tou komvou 1 einai:", th1
 testaki = 0
 meso = 0
 For i = 0 To M Step 1
    ath = 0
    For j = 0 To (M - i) Step 1
        z = M - i - j
        pk(j, i) = (xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ z) / (bi(j) *
bi(z) * G2)
        ath = ath + pk(j, i)
    Next j
    py(i) = ath
    meso = meso + i * py(i)
    testaki = testaki + py(i)
 Next i

 optavereven = th1 * (ps(k) + pns(k)) * r - b * meso - th1 * (1 - ps(k)
- pns(k)) * c
```

```
Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum,
Print #filenum, "O pinakas me tis pithanotites twn katastasewn  p(x,y)
tha einai"
    For x = M To 0 Step -1
Print #filenum, x,
        For y = 0 To (M - x) Step 1                  'Epilogi apofasis
gia kathe katastasi
            Print #filenum, pk(x, y),
        Next y
        Print #filenum, x
        Print #filenum,
    Next x
    Print #filenum, "y",
    For y = 1 To M Step 1
    Print #filenum, y,
    Next y
    Print #filenum, "y"
    Print #filenum,
    Print #filenum, "testaki=", testaki

Next k

End Sub

Sub Ypologismos_G()

Dim i As Integer
Dim j As Integer
Dim z As Integer

ath = 0

For i = 0 To plithos Step 1      'plithos ston komvo 1
        For j = 0 To (plithos - i) Step 1 'plithos ston komvo 2
            z = (plithos - i - j)           'plithos ston komvo 3
            ath = ath + ((xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ z)) /
(bi(j) * bi(z))
        Next j
Next i

End Sub
```

Παράρτημα Α3

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης πολιτικής στο πρόβλημα δύο σταδίων της παραγράφου 2.4.2. Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos
Public x As Double              'taktikoi pelates se troxia
Public y As Double              'pelates sto systima, prwti mixani
Public z As Double              'pelates sto systima, deuteri mixani
'M-x-y-z : oi efimeroi pelates ektos systymatos
Public lev As Integer           'epipeda poititas
Public p() As Double           'dianisma pithanotitwn epipedwn poiotitas
Public s() As Double              'dianisma pithanotitwn ikanopoiisis
pelatwn apo agora proiontwn apo ta epipeda poiotitas
Public lamda(2) As Double         'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi(2) As Double             'rythmos eksypiretisis
Public c As Double              'monadiaio kostos aporripsis kommatiou
Public r As Double              'monadiao ofelos pwlisis proiontos
Public b1 As Double               'monadiaio kostos anamonis ana monada
xronou sto prwto stadio
Public b2 As Double               'monadiaio kostos anamonis ana monada
xronou sto deutero stadio
'Public h As Double                  'monadiaio kostos apothematos sto
endiameso stadio
Public vmax As Double           'megistos synolikos rythmos afiksis
Public vold() As Double        'pinakas me ta esoda gia kathe katastasi
ti xroniki stigmi k
Public vnew() As Double     'pinakas me ta esoda gia kathe katastasi ti
xroniki stigmi k+1
Public dis1() As Integer      'pinakas me ti metavliti apofasis gia tis
antistoixes katastaseis: 1=poulaw, 0=petaw
Public error As Double         'apoklisi kerdwn metaksy twn stigmwn k+1
kai k
Public merror As Double             'orio apoklisis gia to peras twn
epanalipsewn
Public rep As Long           'metritis epanalispewn
Public pen As Double           'penalti, kostos, kakis poiotitas
Public optavereven As Double    'meso veltisto kerdos
Public sign As Integer          'deiktis yparksis arnitikou kerdous

Sub Main()

Dim i As Integer
Dim dif As Double           'Boithitikes metavlites
Dim maxdif As Double
Dim mindif As Double
Dim ath As Double          'metavliti gia to athroisma stous typous twn
V
Dim max1 As Double
```

```
'apodosi arxikwn timwn kai dedomenwn

Open App.Path & "\data.txt" For Input As #1          'Anoigma arxeiou
dedomenwn gia diavasma
 Input #1, M
 Input #1, lev

 ReDim vold(M, M, M)                                 'Epanakoa8orismos
megethous pinakwn
ReDim vnew(M, M, M)
ReDim dis1(M, M, M)
ReDim p(lev)
ReDim s(lev)
 For i = lev To 0 Step -1
 Input #1, p(i)
 Next i
 For i = lev To 0 Step -1
 Input #1, s(i)
 Next i

Input #1, lamda(1)
Input #1, lamda(2)
Input #1, mi(1)
Input #1, mi(2)
Input #1, c
Input #1, r
Input #1, b1
Input #1, b2
Input #1, pen
Input #1, merror


vmax = mi(1) + mi(2) + M * lamda(1)

error = 1000000
rep = 0

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ results.txt" For Output As #filenum       'Anoigma
arxeiou apotelesmatwn

While (error < -merror) Or (error > merror)                 'Synthiki
epanalipsis
 rep = rep + 1
 For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
        For z = 0 To (M - x - y) Step 1   'Ypologismos twn V(k+1) gia
oles tis katastaseis
            If x = 0 Then
                If y = 0 Then
                    If z = 0 Then
                        'KATASTASI (0,0,0)
                        vnew(0, 0, 0) = (1 / vmax) * (M * lamda(2) *
vold(0, 1, 0) + (M * (lamda(1) - lamda(2)) + mi(1) + mi(2)) * vold(0,
0, 0))

'*********************************************************************
****************************************
```

```
                    Else
                        ath = 0
                        For i = 0 To lev Step 1
                            max1 = 0  ' KATASTASI(0,0,z)
                            If (s(i) * vold(1, 0, z - 1) + (1 - s(i)) *
vold(0, 0, z - 1) + r) >= (vold(0, 1, z - 1) - c) Then
                                max1 = s(i) * vold(1, 0, z - 1) + (1 -
s(i)) * vold(0, 0, z - 1) + r
                            Else
                                max1 = vold(0, 1, z - 1) - c
                            End If
                            ath = ath + mi(2) * p(i) * max1
                        Next i
                        vnew(0, 0, z) = (1 / vmax) * ((M - z) * lamda(2)
* vold(0, 1, z) + ath + (mi(1) + M * (lamda(1) - lamda(2)) + z * lamda(2))
* vold(0, 0, z) - b2 * z)

                    End If

'**********************************************************************
****************************************
                Else
                    If z = 0 Then
                        If y = M Then 'KATASTASI (0,M,0)
                            vnew(0, M, 0) = (1 / vmax) * (mi(1) * vold(0,
M - 1, 1) + (mi(2) + M * lamda(1)) * vold(0, M, 0) - b1 * M)
                        Else
                        ' KATASTASI(0,y,0)

                        vnew(0, y, 0) = (1 / vmax) * ((M - y) * lamda(2)
* vold(0, y + 1, 0) + mi(1) * vold(0, y - 1, 1) + (mi(2) + M * (lamda(1)
- lamda(2)) + y * lamda(2)) * vold(0, y, 0) - b1 * y)
                        End If

'**********************************************************************
****************************************
                    ElseIf (y + z) = M Then
                        ath = 0  'KATASTASI(0,y,M-y)
                        For i = 0 To lev Step 1
                            max1 = 0
                            If (s(i) * vold(1, y, M - y - 1) + (1 -
s(i)) * vold(0, y, M - y - 1) + r) >= (vold(0, y + 1, M - y - 1) - c)
Then
                                max1 = s(i) * vold(1, y, M - y - 1) +
(1 - s(i)) * vold(0, y, M - y - 1) + r
                            Else
                                max1 = vold(0, y + 1, M - y - 1) - c
                            End If
                            ath = ath + mi(2) * p(i) * max1
                        Next i
                        vnew(0, y, M - y) = (1 / vmax) * (mi(1) * vold(0,
y - 1, z + 1) + ath + M * lamda(1) * vold(0, y, M - y) - b1 * y - b2 *
(M - y))

'**********************************************************************
****************************************
                    Else
                        ath = 0  'KATASTASI(0,y,z)
                        For i = 0 To lev Step 1
                            max1 = 0
                            If (s(i) * vold(1, y, z - 1) + (1 - s(i)) *
```

```
vold(0, y, z - 1) + r) >= (vold(0, y + 1, z - 1) - c) Then
                                max1 = s(i) * vold(1, y, z - 1) + (1 -
s(i)) * vold(0, y, z - 1) + r
                            Else
                                max1 = vold(0, y + 1, z - 1) - c
                            End If
                            ath = ath + mi(2) * p(i) * max1
                        Next i
                        vnew(0, y, z) = (1 / vmax) * ((M - y - z) *
lamda(2) * vold(0, y + 1, z) + mi(1) * vold(0, y - 1, z + 1) + ath + (M
* (lamda(1) - lamda(2)) + (y + z) * lamda(2)) * vold(0, y, z) - b1 * y
- b2 * z)
                    End If

'*******************************************************************
****************************************
                End If
            Else
                If y = 0 Then
                    If z = 0 Then
                         ' KATASTASI (x,0,0)

                        vnew(x, 0, 0) = (1 / vmax) * (x * lamda(1) *
vold(x - 1, 1, 0) + (M - x) * lamda(2) * vold(x, 1, 0) + ((M - x) *
(lamda(1) - lamda(2)) + mi(1) + mi(2)) * vold(x, 0, 0))

'*******************************************************************
****************************************
                    ElseIf (x + z) = M Then
                        ath = 0  ' KATASTASI (x,0,M-x)
                        For i = 0 To lev Step 1
                            max1 = 0
                            If (s(i) * vold(x + 1, 0, M - x - 1) + (1 -
s(i)) * vold(x, 0, M - x - 1) + r) >= (vold(x, 1, M - x - 1) - c) Then
                                max1 = s(i) * vold(x + 1, 0, M - x - 1)
+ (1 - s(i)) * vold(x, 0, M - x - 1) + r
                            Else
                                max1 = vold(x, 1, M - x - 1) - c
                            End If
                            ath = ath + mi(2) * p(i) * max1
                        Next i
                        vnew(x, 0, M - x) = (1 / vmax) * (x * lamda(1)
* vold(x - 1, 1, M - x) + ath + (mi(1) + (M - x) * lamda(1)) * vold(x,
0, M - x) - (M - x) * b2)

'*******************************************************************
****************************************
                    Else
                        ath = 0  'KATASTASI(x,0,z)
                        For i = 0 To lev Step 1
                            max1 = 0
                            If (s(i) * vold(x + 1, 0, z - 1) + (1 -
s(i)) * vold(x, 0, z - 1) + r) >= (vold(x, 1, z - 1) - c) Then
                                max1 = s(i) * vold(x + 1, 0, z - 1) +
(1 - s(i)) * vold(x, 0, z - 1) + r
                            Else
                                max1 = vold(x, 1, z - 1) - c
                            End If
                            ath = ath + mi(2) * p(i) * max1
                        Next i
                        vnew(x, 0, z) = (1 / vmax) * (x * lamda(1) *
```

```
vold(x - 1, 1, z) + (M - x - z) * lamda(2) * vold(x, 1, z) + ath + (mi(1)
+ (M - x) * (lamda(1) - lamda(2)) + z * lamda(2)) * vold(x, 0, z) - z *
b2)

'*********************************************************************
****************************************
                    End If
                Else
                    If z = 0 Then
                        If x + y = M Then
                              'KATASTASI (x,M-x,0)

                            vnew(x, M - x, 0) = (1 / vmax) * (x *
lamda(1) * vold(x - 1, M - x + 1, 0) + mi(1) * vold(x, M - x - 1, 1) +
(mi(2) + (M - x) * lamda(1)) * vold(x, M - x, 0) - b1 * (M - x))

'*********************************************************************
****************************************
                        Else
                             'KATASTASI (x,y,0)

                            vnew(x, y, 0) = (1 / vmax) * (x * lamda(1)
* vold(x - 1, y + 1, 0) + (M - x - y) * lamda(2) * vold(x, y + 1, 0) +
mi(1) * vold(x, y - 1, 1) + (mi(2) + (M - x) * (lamda(1) - lamda(2)) +
y * lamda(2)) * vold(x, y, 0) - b1 * y)

'*********************************************************************
****************************************
                        End If
                    ElseIf (x + y + z) = M Then
                        ath = 0  'KATASTASI (x,y,M-x-y)
                        For i = 0 To lev Step 1
                            max1 = 0
                            If (s(i) * vold(x + 1, y, M - x - y - 1) +
(1 - s(i)) * vold(x, y, M - x - y - 1) + r) >= (vold(x, y + 1, M - x -
y - 1) - c) Then
                                max1 = s(i) * vold(x + 1, y, M - x - y
- 1) + (1 - s(i)) * vold(x, y, M - x - y - 1) + r
                            Else
                                max1 = vold(x, y + 1, M - x - y - 1) -
c
                            End If
                            ath = ath + mi(2) * p(i) * max1
                        Next i
                        vnew(x, y, M - x - y) = (1 / vmax) * (x * lamda(1)
* vold(x - 1, y + 1, M - x - y) + mi(1) * vold(x, y - 1, M - x - y + 1)
+ ath + (M - x) * lamda(1) * vold(x, y, M - x - y) - b1 * y - b2 * (M -
x - y))

'*********************************************************************
****************************************
                    Else
                        ath = 0  'KATASTASI (x,y,z)
                        For i = 0 To lev Step 1
                            max1 = 0
                            If (s(i) * vold(x + 1, y, z - 1) + (1 -
s(i)) * vold(x, y, z - 1) + r) >= (vold(x, y + 1, z - 1) - c) Then
                                max1 = s(i) * vold(x + 1, y, z - 1) +
(1 - s(i)) * vold(x, y, z - 1) + r
                            Else
                                max1 = vold(x, y + 1, z - 1) - c
```

```
                        End If
                        ath = ath + mi(2) * p(i) * max1
                    Next i
                    vnew(x, y, z) = (1 / vmax) * (x * lamda(1) *
vold(x - 1, y + 1, z) + (M - x - y - z) * lamda(2) * vold(x, y + 1, z)
+ mi(1) * vold(x, y - 1, z + 1) + ath + ((M - x) * (lamda(1) - lamda(2))
+ (y + z) * lamda(2)) * vold(x, y, z) - b1 * y - b2 * z)

'**********************************************************************
****************************************
                End If
            End If
        End If
    Next z
  Next y
Next x


 sign = 1
   maxdif = 0
   mindif = 1000000
   For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
     For z = 0 To (M - x - y) Step 1
        dif = vnew(x, y, z) - vold(x, y, z)          'Diafora sto V
metaksy diadoxikwn xronikwn stigmwn
        If (dif < 0) Then
            dif = -dif
            sign = -1
        Else
            sign = 1
        End If
        If (dif < mindif) Then
            mindif = dif
        End If
        If (dif > maxdif) Then
            maxdif = dif
        End If
     Next z
    Next y
   Next x

   error = maxdif - mindif                     'Kathorismos tis apoklisis


   optavereven = maxdif * vmax                        'Ypologismos mesou
kerdous

   For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
      For z = 0 To (M - x - y) Step 1
        vold(x, y, z) = vnew(x, y, z)               'Antikatastasi twn
V metaksy twn xronikwn stigmwn k kai k+1, gia metabasi se epomeni xroniki
stigmi
      Next z
    Next y
   Next x

Wend                                                'telos epanalipsis

Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
```

```
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print #filenum, "O rythmos afiksis twn eukairiakwn pelatwn einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis tis prwtis mixanis eimai
mi(1)=", mi(1)
Print #filenum, "O rythmos eksypiretisis tis deuteris mixanis eimai
mi(2)=", mi(2)
Print #filenum, "To kostos aporripsis enos kommatiou einai c=", c
Print #filenum, "To kostos anamonis enos pelati ana monada xronou sto
1o stadio einai b1=", b1
Print #filenum, "to kostos apothikeusis enos imikatergasmenou kommatiou
kai anamonis enos pelati sto 2o stadio einai b2=", b2
Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai r=", r
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:",
lev
Print #filenum, "H pithanotita paragwgis kai i pithanotita apodoxis
proiontos einai antistoixa gia ka8e epipedo poiotitas:"
For i = 0 To lev Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s(i)
Next i
Print #filenum,
Print #filenum, "To orio sfalmatos oristike sto ", merror
Print #filenum, "Petyxame sygklisi meta apo arithmo epanalipsewn:", rep
Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum, "sign:", sign
Print #filenum,

For i = 0 To lev Step 1
    Print #filenum, "O pinakas apofasewn pwlisis/aporripsis tha einai
gia to epipedo poiotitas ", i
    For x = 0 To (M - 1) Step 1
    Print #filenum, "O pinakas apofasewn pwlisis/aporripsis tha einai
gia to epipedo plithous taktikwn pelatwn ", x
        For y = (M - x - 1) To 0 Step -1
        Print #filenum, y,
            For z = 1 To (M - x - y) Step 1           'Epilogi apofasis
gia kathe katastasi
                If (s(i) * vold(x + 1, y, z - 1) + (1 - s(i)) * vold(x,
y, z - 1) + r) >= (vold(x, y, z) - c) Then
                    dis1(x, y, z) = 1
                Else
                    dis1(x, y, z) = 0
                End If
                Print #filenum, dis1(x, y, z),
            Next z
            Print #filenum, y
            Print #filenum,
        Next y
        Print #filenum, "z",
        For z = 0 To M Step 1
            Print #filenum, z,
        Next z
        Print #filenum, "z"
        Print #filenum,
    Next x
Next i
End Sub
```

Παράρτημα Α4

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης ευρετικής πολιτικής με την χρήση ΚΔΑ, στο πρόβλημα δύο σταδίων της παραγράφου 2.4.2 Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos
Public x As Double              'taktikoi pelates se troxia
Public y As Double              'pelates sto systima 1o stadio
Public zi As Double             'pelates sto systima 2o stadio
'M-x-y-zi : oi efimeroi pelates ektos systymatos
Public lev As Integer           'epipeda poititas
Public p() As Double          'dianisma pithanotitwn epipedwn poiotitas
Public s() As Double             'dianisma pithanotitwn ikanopoiisis
pelatwn apo agora proiontwn apo ta epipeda poiotitas
Public lamda(2) As Double        'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi(4) As Double             'rythmos eksypiretisis
Public c As Double              'monadiaio kostos aporripsis kommatiou
Public r As Double              'monadiao ofelos pwlisis proiontos
Public b1 As Double               'monadiaio kostos anamonis ana monada
xronou sto 1o stadio
Public b2 As Double                     'monadiaio kostos apothematos
imikatergasmenwn kommatiwn kai anamonis sto 2o stadio
Public bi() As Double             'ta bita gia tous komvous 3,4 (idia)
analoga me to plithos ston komvo
Public u(4) As Double           'ta u gia ka8e komvo
Public ath As Double              ' metritis gia to athroisma ston typo
tou G()
Public plithos As Integer       'to megethos N pou exoume mesa sto G()
Public xi(4) As Double          'ta x gia kathe komvo tou diktiou
Public ps() As Double           'pi8anotita na ikonopoii8ike ananola to
epipedo poiotitas pou exoume san katofli
Public pns() As Double         'pi8anotita na min ikanopoii8ika analoga
me to epipedo pou exoume san katofli
Public optavereven As Double    'meso veltisto kerdos
Public G1 As Double             'G(M-1)
Public G2 As Double             'G(M)
Public th1 As Double            'TH komvou1
Public th2 As Double            'TH komvou2
Public meso1 As Double               'meso plithos pelatwn sto systima
(komvos1)
Public meso2 As Double               'meso plithos pelatwn sto systima
(komvos2)
Public pk() As Double           'Pithanotites katastasewn
Public py1() As Double             'pithanotites plithous pelatwn ston
komvo1
Public py2() As Double             'pithanotites plithous pelatwn ston
komvo2
Public testaki1 As Double       'athroisma pithanotitwn katastasewn....
prepei na vgainei 1
```

```
Public testaki2 As Double

Sub Main()

Dim i As Integer
Dim j As Integer
Dim z As Integer
Dim d As Integer
Dim k As Integer


'apodosi arxikwn timwn kai dedomenwn
Open App.Path & "\data.txt" For Input As #1
 Input #1, M
 Input #1, lev

ReDim bi(M)
ReDim pk(M, M, M)
ReDim py1(M)
ReDim py2(M)

x = 0
y = 0

'epanakathorismos megethos dianismatwn s kai p
ReDim p(lev)
ReDim s(lev)
ReDim ps(lev)
ReDim pns(lev)

'Dedomena gia tis pithanotites
For i = 7 To 0 Step -1
Input #1, p(i)
Next i

For i = lev To 0 Step -1
Input #1, s(i)
Next i

For i = 0 To lev Step 1
Input #1, ps(i)
Next i

For i = 0 To lev Step 1
Input #1, pns(i)
Next i


'Dedomena
Input #1, lamda(1)
Input #1, lamda(2)
Input #1, mi(1)
Input #1, mi(2)
Input #1, c
Input #1, r
Input #1, b1
Input #1, b2


'Ypologismos twn bita gia tous komvous 3,4
'Gia  ton  komvo  1,  ola  ta  bita  einai  =1  afou  exoume  mia  monada
```

```
eksypiretisis
bi(0) = 1
For i = 1 To M Step 1
    bi(i) = i * bi(i - 1)
Next i

mi(3) = lamda(1)
mi(4) = lamda(2)

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ fms_results.txt" For Output As #filenum      'Anoigma
arxeiou apotelesmatwn

Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print #filenum, "O rythmos afiksis twn eukairiakwn pelatwn einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis sto 1o stadio eimai mi(1)=",
mi(1)
Print #filenum, "O rythmos eksypiretisis sto 2o stadio einai mi(2)=",
mi(2)
Print #filenum, "To kostos aporripsis enos kommatiou einai c=", c
Print #filenum, "To kostos anamonis enos pelati ana monada xronou sto
1o stadio einai b1=", b1
Print #filenum, "To kostos apothematos gia ta imikatergasmena kommatia
kai anamonis sto 2o stadio einai b2=", b2
Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai r=", r
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:",
lev + 1
Print #filenum, "H pithanotita paragwgis kai i pithanotita apodoxis gia
kathe epipedo poiotitas einai antistoixa:"
For i = 0 To lev Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s(i)
Next i
Print #filenum,

'Ypologismos twn u gia kathe komvo
For k = 0 To 7 Step 1             'Gia ola ta pithana katoflia apodektou
epipedou poiotitas
    u(1) = 1 / (2 + ps(k) + pns(k))
    u(2) = 1 / (2 + ps(k) + pns(k))
    u(3) = ps(k) / (2 + ps(k) + pns(k))
    u(4) = pns(k) / (2 + ps(k) + pns(k))

    For i = 1 To 4
        xi(i) = u(i) / mi(i)
    Next i

    plithos = M
    Call Ypologismos_G
    G2 = ath
    plithos = M - 1
    Call Ypologismos_G
    G1 = ath

    th1 = u(1) * G1 / G2
```

```
    th2 = u(2) * G1 / G2


 Print #filenum, "An theorisoume san katofli apodektis poiotitas to
epipedo", k
 Print #filenum, "To throughput tou komvou 2 einai:", th2
 testaki1 = 0
 meso1 = 0
 For i = 0 To M Step 1 'komvos1
    ath = 0
    For j = 0 To (M - i) Step 1 'komvos2
        For z = 0 To (M - i - j) Step 1               'komvos3
            d = M - i - j - z       'komvos4
            pk(i, j, z) = ((xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ z) *
(xi(4) ^ d)) / (bi(d) * bi(z) * G2)
            ath = ath + pk(i, j, z)
        Next z
    Next j
    py1(i) = ath
    meso1 = meso1 + i * py1(i)
    testaki1 = testaki1 + py1(i)
 Next i

 testaki2 = 0
 meso2 = 0
 For j = 0 To M Step 1 'komvos2
    ath = 0
    For i = 0 To (M - j) Step 1 'komvos1
        For z = 0 To (M - i - j) Step 1               'komvos3
            d = M - i - j - z       'komvos4
            pk(i, j, z) = ((xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ z) *
(xi(4) ^ d)) / (bi(d) * bi(z) * G2)
            ath = ath + pk(i, j, z)
        Next z
    Next i
    py2(j) = ath
    meso2 = meso2 + j * py2(j)
    testaki2 = testaki2 + py2(j)
 Next j

 optavereven = th2 * (ps(k) + pns(k)) * r - b1 * meso1 - b2 * meso2 -
th2 * (1 - ps(k) - pns(k)) * c


Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum,
Print #filenum, "testaki1=", testaki1
Print #filenum, "testaki2=", testaki2
Next k

End Sub

Sub Ypologismos_G()

Dim i As Integer
Dim j As Integer
Dim z As Integer
Dim d As Integer

ath = 0
```

```
For i = 0 To plithos Step 1      'plithos ston komvo 1
        For j = 0 To (plithos - i) Step 1 'plithos ston komvo 2
            For z = 0 To (plithos - i - j) Step 1 'plithos sto komvo 3
                 d = (plithos - i - j - z)         'plithos ston komvo
4
                ath = ath + ((xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ z) *
(xi(4) ^ d)) / (bi(d) * bi(z))
            Next z
        Next j
Next i

End Sub
```

Παράρτημα Β1

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης πολιτικής στο πρόβλημα του Κεφαλαίου 3. Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos agoras
Public x As Double              'taktikoi pelates sto systima
Public y As Double              'eukairiakoi pelates sto systima
Public z As Double              'taktikoi pel;ates ektos systimatos
'M-x-y-z : oi efimeroi pelates ektos systymatos
Public lev As Integer           'epipeda poititas
Public p() As Double           'dianisma pithanotitwn epipedwn poiotitas
Public s() As Double           'dianisma pithanotitwn ikanopoiisis
pelatwn apo agora proiontwn apo ta epipeda poiotitas
Public lamda(2) As Double        'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi As Double             'rythmos eksypiretisis
Public r As Double              'monadiaio kostos aporripsis kommatiou
Public pr As Double              'monadiao ofelos pwlisis proiontos
Public b1 As Double              'monadiaio kostos anamonis ana monada
xronou taktikou pelati
Public b2 As Double                 'eukairiakou pelati
Public vmax As Double           'megistos synolikos rythmos afiksis
Public vold() As Double       'pinakas me ta esoda gia kathe katastasi
ti xroniki stigmi k
Public vnew() As Double     'pinakas me ta esoda gia kathe katastasi ti
xroniki stigmi k+1
Public dis() As Integer      'pinakas me ti metavliti apofasis gia tis
antistoixes katastaseis: 1=poulaw se taktiko, 2=poulaw se eukairiako,
0=petaw
Public error As Double         'apoklisi kerdwn metaksy twn stigmwn k+1
kai k
Public merror As Double            'orio apoklisis gia to peras twn
epanalipsewn
Public rep As Long           'metritis epanalispewn
Public pen As Double           'penalti, kostos, kakis poiotitas
Public optavereven As Double    'meso veltisto kerdos
Public sign As Integer          'deiktis yparksis arnitikou kerdous

Sub main()

Dim i As Integer
Dim dif As Double           'Boithitikes metavlites
Dim maxdif As Double
Dim mindif As Double
Dim ath As Double          'metavliti gia to athroisma stous typous twn
V
Dim end1 As Double         'Endyktikes synartiseis I(x>0) epi to analogo
Dim end2 As Double          'I(y>0) epi to analogo
Dim end3 As Double          'I(x+y>0) epi to analogo
```

```
Dim end3tonos As Double     'I(x+y>0) mono h endeiktiki
Dim max As Double
Dim protosoros As Double
Dim deuterosoros As Double


'apodosi arxikwn timwn kai dedomenwn

Open App.Path & "\data.txt" For Input As #1          'Anoigma arxeiou
dedomenwn gia diavasma
 Input #1, M
 Input #1, lev
 ReDim vold(M, M, M)                                 'Epanakoa8orismos
megethous pinakwn
ReDim vnew(M, M, M)
ReDim dis(M, M, M)
ReDim p(lev)
ReDim s(2, lev)
For i = lev To 1 Step -1
 Input #1, p(i)
Next i
For i = lev To 1 Step -1
 Input #1, s(1, i)
 Input #1, s(2, i)
Next i

Input #1, lamda(1)
Input #1, lamda(2)
Input #1, mi
Input #1, r
Input #1, pr
Input #1, b1
Input #1, b2
Input #1, pen
Input #1, merror


vmax = mi + M * lamda(1)

error = 1000000
rep = 0

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ results.txt" For Output As #filenum       'Anoigma
arxeiou apotelesmatwn

While (error < -merror) Or (error > merror)                 'Synthiki
epanalipsis
 rep = rep + 1
 For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1                          'Ypologismos
twn V(k+1) gia oles tis katastaseis
        For z = 0 To (M - x - y) Step 1
            ath = 0
            For i = 0 To lev Step 1
                If x > 0 Then
                    end1 = (s(1, i) * vold(x - 1, y, z + 1) + (1 - s(1,
i)) * vold(x - 1, y, z) + pr)
                Else
```

```
                end1 = 0
            End If
            If y > 0 Then
                end2 = (s(2, i) * vold(x, y - 1, z + 1) + (1 - s(2,
i)) * vold(x, y - 1, z) + pr)
            Else
                end2 = 0
            End If
            If (x + y) > 0 Then
                end3 = (vold(x, y, z) - r)
                end3tonos = 1
            Else
                end3 = 0
                end3tonos = 0
            End If
            If end1 >= end2 And end1 >= end3 Then
                max = end1
            ElseIf end2 > end1 And end2 > end3 Then
                max = end2
            Else
                max = end3
            End If
            ath = ath + mi * p(i) * max
        Next i
        If z = 0 Or x = M Then
            protosoros = 0
        Else
            protosoros = z * lamda(1) * vold(x + 1, y, z - 1)
        End If
        If (x + y + z) = M Then
            deuterosoros = 0
        Else
            deuterosoros = (M - x - y - z) * lamda(2) * vold(x, y +
1, z)
        End If
        vnew(x, y, z) = (1 / vmax) * (protosoros + deuterosoros +
ath + ((M - z) * (lamda(1) - lamda(2)) + (x + y) * lamda(2) + (1 -
end3tonos) * mi) * vold(x, y, z) - (x * b1 + y * b2))
        Next z
    Next y
Next x


sign = 1
maxdif = 0
mindif = 1000000
For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
        For z = 0 To (M - x - y) Step 1

            dif = vnew(x, y, z) - vold(x, y, z)          'Diafora sto V
metaksy diadoxikwn xronikwn stigmwn
            If (dif < 0) Then
                dif = -dif
                sign = -1
            Else
                sign = 1
            End If
            If (dif < mindif) Then
                mindif = dif
            End If
```

```
            If (dif > maxdif) Then
                maxdif = dif
            End If
        Next z
    Next y
Next x

error = maxdif - mindif                      'Kathorismos tis apoklisis


optavereven = maxdif * vmax                  'Ypologismos mesou kerdous

For x = 0 To M Step 1
    For y = 0 To (M - x) Step 1
        For z = 0 To (M - x - y) Step 1
            vold(x, y, z) = vnew(x, y, z)              'Antikatastasi
twn V metaksy twn xronikwn stigmwn k kai k+1, gia metabasi se epomeni
xroniki stigmi
        Next z
    Next y
Next x

Wend                                            'telos epanalipsis

Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print  #filenum,  "O  rythmos  afiksis  twn  eukairiakwn  pelatwn  einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis eimai mi=", mi
Print #filenum, "To kostos aporripsis enos kommatiou einai r=", r
Print #filenum, "To kostos anamonis enos pelati ana monada xronou einai,
gia taktikous kai eukairiakous antistoixa:", b1, ",", b2
Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai pr=", pr
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:",
lev
Print   #filenum,   "H   pithanotita   paragwgis   kai   oi   pithanotites
ikanopoiisis enos taktikou kai enos eukairiakou pelati"
Print #filenum, "gia kathe epipedo poiotitas einai antistoixa:"
For i = 1 To lev Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s(1, i), ",",
s(2, i)
Next i
Print #filenum,
Print #filenum, "To orio sfalmatos oristike sto ", merror
Print #filenum, "Petyxame sygklisi meta apo arithmo epanalipsewn:", rep
Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum, "sign:", sign
Print #filenum,
For i = 1 To lev Step 1
    Print  #filenum,  "O  pinakas  apofasewn  tha  einai  gia  to  epipedo
poiotitas ", i
    For z = 0 To M Step 1
        Print #filenum, "Gia plithos taktikwn pelatwn ektos systimatos,
z=", z
        For x = (M - z) To 0 Step -1
            Print #filenum, x,
            For y = 0 To (M - x - z) Step 1                    'Epilogi
apofasis gia kathe katastasi
```

```
                If x > 0 Then
                    end1 = (s(1, i) * vold(x - 1, y, z + 1) + (1 - s(1,
i)) * vold(x - 1, y, z) + pr)
                Else
                    end1 = 0
                End If
                If y > 0 Then
                    end2 = (s(2, i) * vold(x, y - 1, z + 1) + (1 - s(2,
i)) * vold(x, y - 1, z) + pr)
                Else
                    end2 = 0
                End If
                If (x + y) > 0 Then
                    end3 = (vold(x, y, z) - r)
                Else
                    end3 = 0
                End If
                If end1 >= end2 And end1 >= end3 Then
                    dis(x, y, z) = 1
                ElseIf end2 > end1 And end2 > end3 Then
                    dis(x, y, z) = 2
                Else
                    dis(x, y, z) = 3
                End If
                Print #filenum, dis(x, y, z),
            Next y
            Print #filenum, x
            Print #filenum,
        Next x
        Print #filenum, "y",
        For y = 0 To (M - z) Step 1
            Print #filenum, y,
        Next y
        Print #filenum, "y"
        Print #filenum,
    Next z
Next i


End Sub
```

Παράρτημα Β2

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης ευρετικής πολιτικής με την χρήση ΚΔΑ, στο πρόβλημα του Κεφαλαίου 3. Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos
Public x As Double              'taktikoi pelates se troxia
Public y As Double              'taktikoi pelates sto systima
Public zi As Double             'eukairiakoi pelates sto systima
'Public w As Double              'pelates sto 3o stadio
'M-x-y-zi : oi efimeroi pelates ektos systymatos
Public lev As Integer           'epipeda poititas
Public p() As Double          'dianisma pithanotitwn epipedwn poiotitas
Public s1() As Double              'dianisma pithanotitwn ikanopoiisis
pelatwn apo agora proiontwn apo ta epipeda poiotitas
Public s2() As Double
Public lamda(2) As Double         'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi(3) As Double            'rythmos eksypiretisis
Public c As Double              'monadiaio kostos aporripsis kommatiou
Public r As Double              'monadiao ofelos pwlisis proiontos
Public b1 As Double             'monadiaio kostos anamonis taktikwn ana
monada xronou
Public b2 As Double              'monadiaio kostos anamonis eukairiakwn

Public bi() As Double             'ta bita gia tous komvous 3,4 (idia)
analoga me to plithos ston komvo
Public uc(3, 2) As Double          'ta u twn klassewn
Public u(3) As Double           'ta u gia ka8e komvo
Public ath As Double              ' metritis gia to athroisma ston typo
tou G()
Public plithos As Integer       'to megethos N pou exoume mesa sto G()
Public xi(3) As Double          'ta x gia kathe komvo tou diktiou
Public xic1 As Double
Public xic2 As Double
Public ps1() As Double            'pi8anotita na ikonopoii8ike taktikos
ananola to epipedo poiotitas pou exoume san katofli
Public ps2() As Double
Public pns1() As Double             'pi8anotita na min ikanopoii8ika
taktikos analoga me to epipedo pou exoume san katofli
Public pns2() As Double
Public optavereven As Double    'meso veltisto kerdos
Public G1 As Double             'G(M-1)
Public G2 As Double             'G(M)
Public th1 As Double            'TH komvou1
Public th2 As Double            'TH komvou2
Public th3 As Double            'TH komvou3
Public th3c1 As Double
Public th3c2 As Double
Public mesoc1mesa As Double          'meso plithos taktikwn pelatwn sto
```

```
systima
Public mesoc2mesa As Double           'meso plithos eukairiakwn pelatwn
sto systima
Public meso3 As Double              'meso plithos pelatwn sto systima
(komvos3)
Public mesotekso As Double
Public mesoeuekso As Double
Public pk() As Double           'Pithanotites katastasewn

Public py3() As Double          'pithanotites plithous pelatwn ston 3
Public testaki1 As Double       'athroisma pithanotitwn katastasewn....
prepei na vgainei 1


Sub Main()

Dim i As Integer    'komvos1
Dim j As Integer    'komvos2
Dim z As Integer    'komvos3 taktikoi
Dim d As Integer    'komvos3 synolika
Dim f As Integer    'komvos3 eukairiakoi
Dim k As Integer


'apodosi arxikwn timwn kai dedomenwn
Open App.Path & "\data.txt" For Input As #1
 Input #1, M
 'M=40
 Input #1, lev
'lev=3
ReDim bi(M)
ReDim pk(M, M, M)
ReDim py1(M)
ReDim py2(M)
ReDim py3(M)

x = 0
y = 0

'epanakathorismos megethos dianismatwn s kai p
ReDim p(lev)
ReDim s1(lev)
ReDim s2(lev)
ReDim ps1(lev)
ReDim pns1(lev)
ReDim ps2(lev)
ReDim pns2(lev)

'Dedomena gia tis pithanotites
For i = lev To 1 Step -1
Input #1, p(i)
Next i

For i = lev To 1 Step -1
Input #1, s1(i)
Next i

For i = lev To 1 Step -1
Input #1, s2(i)
Next i
```

```
For i = 1 To lev Step 1
Input #1, ps1(i)
Next i

For i = 1 To lev Step 1
Input #1, pns1(i)
Next i


For i = 1 To lev Step 1
Input #1, ps2(i)
Next i

For i = 1 To lev Step 1
Input #1, pns2(i)
Next i


'Dedomena
Input #1, lamda(1)

Input #1, lamda(2)

Input #1, mi(3)

Input #1, c

Input #1, r

Input #1, b1

Input #1, b2

'Ypologismos twn bita gia tous komvous 1,2
'Gia ton komvo3 ola ta bita einai =1 afou exoume mia monada eksypiretisis
bi(0) = 1
For i = 1 To M Step 1
    bi(i) = i * bi(i - 1)
Next i

mi(1) = lamda(1)
mi(2) = lamda(2)

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ fms_results.txt" For Output As #filenum      'Anoigma
arxeiou apotelesmatwn



Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print #filenum, "O rythmos afiksis twn eukairiakwn pelatwn einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis sto systima eimai mi(3)=", mi(3)

Print #filenum, "To kostos aporripsis enos kommatiou einai c=", c
```

```
Print #filenum, "To kostos anamonis enos taktikou pelati ana monada xronou einai b1=", b1
Print #filenum, "To kostos anamonis enos eukairiakou einai b2=", b2

Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai r=", r
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:", lev
Print #filenum, "H pithanotita paragwgis kai i pithanotita apodoxis gia kathe epipedo poiotitas   gia taktikous kai eukairiakous einai antistoixa:"
For i = 1 To lev Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s1(i), ",", s2(i)
Next i
Print #filenum,

'Ypologismos twn u gia kathe komvo
For k = 1 To 3 Step 1             'Gia ola ta pithana katoflia apodektou epipedou poiotitas

    uc(1, 1) = 1
    uc(3, 2) = pns1(k) / ((ps1(k) + pns1(k)) * ps2(k))
    uc(2, 2) = (pns1(k) / (ps1(k) + pns1(k))) + ((pns2(k) * pns1(k)) / (ps2(k) * (ps1(k) + pns1(k))))
    uc(3, 1) = 1 / (ps1(k) + pns1(k))
    u(1) = uc(1, 1)
    u(2) = uc(2, 2)
    u(3) = uc(3, 1) + uc(3, 2)

    For i = 1 To 3
        xi(i) = u(i) / mi(i)
    Next i

    plithos = M
    Call Ypologismos_G
    G2 = ath
    plithos = M - 1
    Call Ypologismos_G
    G1 = ath

    th1 = u(1) * G1 / G2
    th2 = u(2) * G1 / G2
    th3 = u(3) * G1 / G2

 Print #filenum, "An theorisoume san katofli apodektis poiotitas to epipedo", k
 Print #filenum, "To throughput tou komvou 3 einai:", th3
 testaki1 = 0
 meso3 = 0
 For d = 0 To M Step 1 'komvos3
    ath = 0
    For i = 0 To (M - d) Step 1  'taktikoi ekso
            j = (M - i - d)        'eukairiakoi eksw
            pk(i, j, d) = ((xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ d)) / (bi(i) * bi(j) * G2)
            ath = ath + pk(i, j, d)
    Next i
    py3(d) = ath
    meso3 = meso3 + d * py3(d)
    testaki1 = testaki1 + py3(d)
 Next d
```

```
 mesotekso = th1 / mi(1)
 mesoeuekso = th2 / mi(2)
 th3c1 = th3 * uc(3, 1) / u(3)
 th3c2 = th3 * uc(3, 2) / u(3)
 mesoc1mesa = meso3 * uc(3, 1) / u(3)
 mesoc2mesa = meso3 * uc(3, 2) / u(3)

 optavereven = th3c1 * (ps1(k) + pns1(k)) * r + th3c2 * (ps2(k) + pns2(k))
* r - b1 * mesoc1mesa - b2 * mesoc2mesa - th3c1 * (1 - ps1(k) - pns1(k))
* c - th3c2 * (1 - ps2(k) - pns2(k)) * c


Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum,

    Print #filenum,
    Print #filenum, "testaki1=", testaki1
    Print #filenum, "meso3=", meso3
    Print #filenum, "mesoc1mesa=", mesoc1mesa
    Print #filenum, "mesoc2mesa=", mesoc2mesa
    Print #filenum, "mesotekso=", mesotekso
    Print #filenum, "mesoeuekso=", mesoeuekso
    Print #filenum,
Next k

End Sub

Sub Ypologismos_G()

Dim i As Integer
Dim j As Integer
Dim d As Integer

ath = 0

For i = 0 To plithos Step 1      'plithos ston komvo 1
        For j = 0 To (plithos - i) Step 1 'plithos ston komvo 2
            d = plithos - i - j 'plithos ston komvo 3
            ath = ath + ((xi(1) ^ i) * (xi(2) ^ j) * (xi(3) ^ d)) /
(bi(i) * bi(j))
        Next j
Next i

End Sub
```

Παράρτημα Γ

Παρατίθεται το πρόγραμμα που αναπτύχθηκε για τον υπολογισμό της βέλτιστης πολιτικής στο πρόβλημα του Κεφαλαίου 4. Το πρόγραμμα είναι σε γλώσσα προγραμματισμού Visual Basic.

```
Option Explicit

Public M As Double              'plithismos
Public x As Double              'apothema typou 1
Public y As Double              'apothema typou 2
Public z As Double              'plithos taktikwn pelatwn
Public lev As Integer           'epipeda poititas paragwmenou proiontos
Public p() As Double          'dianisma pithanotitwn epipedwn poiotitas
Public s() As Double              'dianisma pithanotitwn ikanopoiisis
pelatwn typou i apo agora proiontwn epipedou poiotitas j
Public lamda(2) As Double         'dianisma me tous rythmous afiksis:
lamda(1) apo taktikous kai lamda(2) apo efimerous, to lamda(0) den
xrisimopoieitai
Public mi As Double             'rythmos eksypiretisis
Public c As Double              'monadiaio kostos aporripsis kommatiou
Public r As Double              'monadiao ofelos pwlisis proiontos
Public h As Double                'monadiaio kostos anamonis ana monada
xronou
Public vmax As Double           'megistos synolikos rythmos afiksis
Public vold() As Double       'pinakas me ta esoda gia kathe katastasi
ti xroniki stigmi k
Public vnew() As Double     'pinakas me ta esoda gia kathe katastasi ti
xroniki stigmi k+1
Public dis1() As Integer     'pinakas me ti metavliti apofasis paragwgis
gia tis antistoixes katastaseis: 1=paragw, 0=stamataw na paragw
Public dis2() As Integer        'pinakas apofasewn pwlisis se taktikous
pelates: 1=poulaw poiotita 1, 2=poulaw poiotita 2
Public dis3() As Integer      'pinakas apofasewn pwlisis se eukairiakous
pelates: 0=aporiptw zitisi, 1=poulaw poiotita 1, 2=poulaw poiotita 2
Public Kapa As Double
Public K As Double


Public error As Double          'apoklisi kerdwn metaksy twn stigmwn k+1
kai k
Public merror As Double              'orio apoklisis gia to peras twn
epanalipsewn
Public rep As Long            'metritis epanalispewn
Public pen As Double            'penalti, kostos, kakis poiotitas
Public optavereven As Double    'meso veltisto kerdos
Public sign As Integer          'deiktis yparksis arnitikou kerdous
Public max1 As Double
Public max2 As Double
Public max3 As Double


Sub main()

Dim i As Integer
```

```
Dim dif As Double           'Boithitikes metavlites
Dim maxdif As Double
Dim mindif As Double
Dim ath As Double          'metavliti gia to athroisma stous typous twn
V

'apodosi arxikwn timwn kai dedomenwn

Open App.Path & "\data.txt" For Input As #1          'Anoigma arxeiou
dedomenwn gia diavasma
 Input #1, M
 Input #1, lev

ReDim p(lev)
ReDim s(2, lev)
 For i = lev To 1 Step -1
 Input #1, p(i)
 Next i
 For i = (lev - 1) To 1 Step -1
 Input #1, s(1, i)
 Input #1, s(2, i)
 Next i

Input #1, lamda(1)
Input #1, lamda(2)
Input #1, mi
Input #1, c
Input #1, r
Input #1, h
Input #1, pen
Input #1, merror

Kapa = M * lamda(1) * r / h
K = Int(Kapa)

ReDim vold(K, K, M)                          'Epanakoa8orismos megethous
pinakwn
ReDim vnew(K, K, M)
ReDim dis1(K, K, M)
ReDim dis2(K, K, M)
ReDim dis3(K, K, M)

vmax = mi + M * lamda(1)

error = 1000000
rep = 0

Dim filenum%    'orismos arxeiou apotelesmatwn

filenum = FreeFile
Open App.Path & "\ results.txt" For Output As #filenum       'Anoigma
arxeiou apotelesmatwn

While (error < -merror) Or (error > merror)                 'Synthiki
epanalipsis
 rep = rep + 1
 For x = 0 To K Step 1
    For y = 0 To (K - x) Step 1
        For z = 0 To M Step 1              'Ypologismos twn V(k+1) gia
oles tis katastaseis
            If x = 0 Then
```

```
                If y = 0 Then
                    If z = 0 Then
                        'KATASTASI (0,0,0)
                        max1 = 0
                        If (p(3) * (vold(0, 0, 0) - c) + p(1) * vold(1,
0, 0) + p(2) * vold(0, 1, 0)) >= vold(0, 0, 0) Then
                            max1 = p(3) * (vold(0, 0, 0) - c) + p(1) *
vold(1, 0, 0) + p(2) * vold(0, 1, 0)
                        Else
                            max1 = vold(0, 0, 0)
                        End If
                        vnew(0, 0, 0) = (1 / vmax) * (mi * max1 + M *
lamda(1) * vold(0, 0, 0))

'*******************************************************************
***************************************
                    ElseIf z = M Then
                        max1 = 0  '(0,0,M)
                        If (p(3) * (vold(0, 0, M) - c) + p(1) * vold(1,
0, M) + p(2) * vold(0, 1, M)) >= vold(0, 0, M) Then
                            max1 = p(3) * (vold(0, 0, M) - c) + p(1) *
vold(1, 0, M) + p(2) * vold(0, 1, M)
                        Else
                            max1 = vold(0, 0, M)
                        End If
                        vnew(0, 0, M) = (1 / vmax) * (mi * max1 + M *
lamda(1) * vold(0, 0, M - 1))

'*******************************************************************
***************************************
                    Else
                        max1 = 0  '(0,0,z)
                        If (p(3) * (vold(0, 0, z) - c) + p(1) * vold(1,
0, z) + p(2) * vold(0, 1, z)) >= vold(0, 0, z) Then
                            max1 = p(3) * (vold(0, 0, z) - c) + p(1) *
vold(1, 0, z) + p(2) * vold(0, 1, z)
                        Else
                            max1 = vold(0, 0, z)
                        End If
                        vnew(0, 0, z) = (1 / vmax) * (mi * max1 + z *
lamda(1) * vold(0, 0, z - 1) + (M - z) * lamda(1) * vold(0, 0, z))
                    End If

'*******************************************************************
***************************************
                ElseIf y = K Then
                    If z = 0 Then
                        max3 = 0 '(0,K,0)
                        If (s(2, 2) * vold(0, K - 1, 1) + (1 - s(2, 2))
* vold(0, K - 1, 0) + r) >= vold(0, K, 0) Then
                            max3 = s(2, 2) * vold(0, K - 1, 1) + (1 -
s(2, 2)) * vold(0, K - 1, 0) + r
                        Else
                            max3 = vold(0, K, 0)
                        End If
                        vnew(0, K, 0) = (1 / vmax) * (M * lamda(2) *
max3 + (mi + M * (lamda(1) - lamda(2))) * vold(0, K, 0) - K * h)

'*******************************************************************
********************************
                    ElseIf z = M Then
```

```
                    max2 = 0 '(0,K,M)
                    If (s(1, 2) * vold(0, K - 1, M) + (1 - s(1, 2))
* vold(0, K - 1, M - 1) + r) >= vold(0, K, M - 1) Then
                        max2 = s(1, 2) * vold(0, K - 1, M) + (1 -
s(1, 2)) * vold(0, K - 1, M - 1) + r
                    Else
                        max2 = vold(0, K, M - 1)
                    End If
                    vnew(0, K, M) = (1 / vmax) * (M * lamda(1) *
max2 + mi * vold(0, K, M) - K * h)

'*******************************************************************
*********************************
                Else
                    max2 = 0    '(0,K,z)
                    max3 = 0
                    If (s(1, 2) * vold(0, K - 1, z) + (1 - s(1, 2))
* vold(0, K - 1, z - 1) + r) >= vold(0, K, z - 1) Then
                        max2 = s(1, 2) * vold(0, K - 1, z) + (1 -
s(1, 2)) * vold(0, K - 1, z - 1) + r
                    Else
                        max2 = vold(0, K, z - 1)
                    End If
                    If (s(2, 2) * vold(0, K - 1, z + 1) + (1 - s(2,
2)) * vold(0, K - 1, z) + r) >= vold(0, K, z) Then
                        max3 = s(2, 2) * vold(0, K - 1, z + 1) + (1
- s(2, 2)) * vold(0, K - 1, z) + r
                    Else
                        max3 = vold(0, K, z)
                    End If
                    vnew(0, K, z) = (1 / vmax) * (z * lamda(1) *
max2 + (M - z) * lamda(2) * max3 + (mi + (M - z) * (lamda(1) - lamda(2)))
* vold(0, K, z) - K * h)
                End If

'*******************************************************************
************************************
            Else
                If z = 0 Then
                    max1 = 0  '(0,y,0)
                    max3 = 0
                    If (p(3) * (vold(0, y, 0) - c) + p(1) * vold(1,
y, 0) + p(2) * vold(0, y + 1, 0)) >= vold(0, y, 0) Then
                        max1 = p(3) * (vold(0, y, 0) - c) + p(1) *
vold(1, y, 0) + p(2) * vold(0, y + 1, 0)
                    Else
                        max1 = vold(0, y, 0)
                    End If
                    If (s(2, 2) * vold(0, y - 1, 1) + (1 - s(2, 2))
* vold(0, y - 1, 0) + r) >= vold(0, y, 0) Then
                        max3 = s(2, 2) * vold(0, y - 1, 1) + (1 -
s(2, 2)) * vold(0, y - 1, 0) + r
                    Else
                        max3 = vold(0, y, 0)
                    End If
                    vnew(0, y, 0) = (1 / vmax) * (mi * max1 + M *
lamda(2) * max3 + M * (lamda(1) - lamda(2)) * vold(0, y, 0) - h * y)

'*******************************************************************
****************************************
                ElseIf z = M Then
```

```
                        max1 = 0  '(0,y,M)
                        max2 = 0
                        If (p(3) * (vold(0, y, M) - c) + p(1) * vold(1,
y, M) + p(2) * vold(0, y + 1, M)) >= vold(0, y, M) Then
                            max1 = p(3) * (vold(0, y, M) - c) + p(1) *
vold(1, y, M) + p(2) * vold(0, y + 1, M)
                        Else
                            max1 = vold(0, y, M)
                        End If
                        If (s(1, 2) * vold(0, y - 1, M) + (1 - s(1, 2))
* vold(0, y - 1, M - 1) + r) >= vold(0, y, M - 1) Then
                            max2 = s(1, 2) * vold(0, y - 1, M) + (1 -
s(1, 2)) * vold(0, y - 1, M - 1) + r
                        Else
                            max2 = vold(0, y, M - 1)
                        End If
                        vnew(0, y, M) = (1 / vmax) * (mi * max1 + M *
lamda(1) * max2 - h * y)

'*********************************************************************
****************************************
                    Else
                        max1 = 0  '(0,y,z)
                        max2 = 0
                        max3 = 0
                        If (p(3) * (vold(0, y, z) - c) + p(1) * vold(1,
y, z) + p(2) * vold(0, y + 1, z)) >= vold(0, y, z) Then
                            max1 = p(3) * (vold(0, y, z) - c) + p(1) *
vold(1, y, z) + p(2) * vold(0, y + 1, z)
                        Else
                            max1 = vold(0, y, z)
                        End If
                        If (s(1, 2) * vold(0, y - 1, z) + (1 - s(1, 2))
* vold(0, y - 1, z - 1) + r) >= vold(0, y, z - 1) Then
                            max2 = s(1, 2) * vold(0, y - 1, z) + (1 -
s(1, 2)) * vold(0, y - 1, z - 1) + r
                        Else
                            max2 = vold(0, y, z - 1)
                        End If
                        If (s(2, 2) * vold(0, y - 1, z + 1) + (1 - s(2,
2)) * vold(0, y - 1, z) + r) >= vold(0, y, z) Then
                            max3 = s(2, 2) * vold(0, y - 1, z + 1) + (1
- s(2, 2)) * vold(0, y - 1, z) + r
                        Else
                            max3 = vold(0, y, z)
                        End If
                        vnew(0, y, z) = (1 / vmax) * (mi * max1 + z *
lamda(1) * max2 + (M - z) * lamda(2) * max3 + (M - z) * (lamda(1) -
lamda(2)) * vold(0, y, z) - h * y)
                    End If

'*********************************************************************
****************************************
                End If
            ElseIf x = K Then
                If z = 0 Then
                    max3 = 0 '(K,0,0)
                    If (s(2, 1) * vold(K - 1, 0, 1) + (1 - s(2, 1)) *
vold(K - 1, 0, 0) + r) >= vold(K, 0, 0) Then
                        max3 = s(2, 1) * vold(K - 1, 0, 1) + (1 - s(2,
1)) * vold(K - 1, 0, 0) + r
```

```
                    Else
                        max3 = vold(K, 0, 0)
                    End If
                    vnew(K, 0, 0) = (1 / vmax) * (M * lamda(2) * max3 +
(mi + M * (lamda(1) - lamda(2))) * vold(K, 0, 0) - K * h)

'*******************************************************************
************************************
                ElseIf z = M Then
                            '(K,0,M)
                    vnew(K, 0, M) = (1 / vmax) * (M * lamda(1) * (s(1,
1) * vold(K - 1, 0, M) + (1 - s(1, 1)) * vold(K - 1, 0, M - 1) + r) +
mi * vold(K, 0, M) - K * h)

'*******************************************************************
************************************
                Else
                    max3 = 0 '(K,0,z)
                    If (s(2, 1) * vold(K - 1, 0, z + 1) + (1 - s(2, 1))
* vold(K - 1, 0, z) + r) >= vold(K, 0, z) Then
                        max3 = s(2, 1) * vold(K - 1, 0, z + 1) + (1 -
s(2, 1)) * vold(K - 1, 0, z) + r
                    Else
                        max3 = vold(K, 0, z)
                    End If
                    vnew(K, 0, z) = (1 / vmax) * (z * lamda(1) * (s(1,
1) * vold(K - 1, 0, z) + (1 - s(1, 1)) * vold(K - 1, 0, z - 1) + r) +
(M - z) * lamda(2) * max3 + (mi + (M - z) * (lamda(1) - lamda(2))) *
vold(K, 0, z) - K * h)

'*******************************************************************
*************************************
                End If
            Else
                If y = 0 Then
                    If z = 0 Then
                        max1 = 0  '(x,0,0)
                        max3 = 0
                        If (p(3) * (vold(x, 0, 0) - c) + p(1) * vold(x
+ 1, 0, 0) + p(2) * vold(x, 1, 0)) >= vold(x, 0, 0) Then
                            max1 = p(3) * (vold(x, 0, 0) - c) + p(1) *
vold(x + 1, 0, 0) + p(2) * vold(x, 1, 0)
                        Else
                            max1 = vold(x, 0, 0)
                        End If
                        If (s(2, 1) * vold(x - 1, 0, 1) + (1 - s(2, 1))
* vold(x - 1, 0, 0) + r) >= vold(x, 0, 0) Then
                            max3 = s(2, 1) * vold(x - 1, 0, 1) + (1 -
s(2, 1)) * vold(x - 1, 0, 0) + r
                        Else
                            max3 = vold(x, 0, 0)
                        End If
                        vnew(x, 0, 0) = (1 / vmax) * (mi * max1 + M *
lamda(2) * max3 + M * (lamda(1) - lamda(2)) * vold(x, 0, 0) - h * x)

'*******************************************************************
***************************************
                    ElseIf z = M Then
                        max1 = 0  '(x,0,M)
                        If (p(3) * (vold(x, 0, M) - c) + p(1) * vold(x
+ 1, 0, M) + p(2) * vold(x, 1, M)) >= vold(x, 0, M) Then
```

```
                        max1 = p(3) * (vold(x, 0, M) - c) + p(1) *
vold(x + 1, 0, M) + p(2) * vold(x, 1, M)
                    Else
                        max1 = vold(x, 0, M)
                    End If
                    vnew(x, 0, M) = (1 / vmax) * (mi * max1 + M *
lamda(1) * (s(1, 1) * vold(x - 1, 0, M) + (1 - s(1, 1)) * vold(x - 1,
0, M - 1) + r) - h * x)

'*********************************************************************
***************************************
                Else
                    max1 = 0  '(x,0,z)
                    max3 = 0
                    If (p(3) * (vold(x, 0, z) - c) + p(1) * vold(x
+ 1, 0, z) + p(2) * vold(x, 1, z)) >= vold(x, 0, z) Then
                        max1 = p(3) * (vold(x, 0, z) - c) + p(1) *
vold(x + 1, 0, z) + p(2) * vold(x, 1, z)
                    Else
                        max1 = vold(x, 0, z)
                    End If
                    If (s(2, 1) * vold(x - 1, 0, z + 1) + (1 - s(2,
1)) * vold(x - 1, 0, z) + r) >= vold(x, 0, z) Then
                        max3 = s(2, 1) * vold(x - 1, 0, z + 1) + (1
- s(2, 1)) * vold(x - 1, 0, z) + r
                    Else
                        max3 = vold(x, 0, z)
                    End If
                    vnew(x, 0, z) = (1 / vmax) * (mi * max1 + z *
lamda(1) * (s(1, 1) * vold(x - 1, 0, z) + (1 - s(1, 1)) * vold(x - 1,
0, z - 1) + r) + (M - z) * lamda(2) * max3 + (M - z) * (lamda(1) -
lamda(2)) * vold(x, 0, z) - h * x)

'*********************************************************************
***************************************
                End If
            ElseIf (x + y) = K Then
                If z = 0 Then
                    max3 = 0    '(x,y,0) (x+y=K)
                    If (s(2, 1) * vold(x - 1, y, 1) + (1 - s(2, 1))
* vold(x - 1, y, 0) + r) >= vold(x, y, 0) And (s(2, 1) * vold(x - 1, y,
1) + (1 - s(2, 1)) * vold(x - 1, y, 0) + r) >= (s(2, 2) * vold(x, y -
1, 1) + (1 - s(2, 2)) * vold(x, y - 1, 0) + r) Then
                        max3 = s(2, 1) * vold(x - 1, y, 1) + (1 -
s(2, 1)) * vold(x - 1, y, 0) + r
                    ElseIf (s(2, 2) * vold(x, y - 1, 1) + (1 - s(2,
2)) * vold(x, y - 1, 0) + r) >= (s(2, 1) * vold(x - 1, y, 1) + (1 - s(2,
1)) * vold(x - 1, y, 0) + r) And (s(2, 2) * vold(x, y - 1, 1) + (1 -
s(2, 2)) * vold(x, y - 1, 0) + r) >= vold(x, y, 0) Then
                        max3 = s(2, 2) * vold(x, y - 1, 1) + (1 -
s(2, 2)) * vold(x, y - 1, 0) + r
                    ElseIf vold(x, y, 0) >= (s(2, 1) * vold(x - 1,
y, 1) + (1 - s(2, 1)) * vold(x - 1, y, 0) + r) And vold(x, y, 0) >=
(s(2, 2) * vold(x, y - 1, 1) + (1 - s(2, 2)) * vold(x, y - 1, 0) + r)
Then
                        max3 = vold(x, y, 0)
                    End If
                    vnew(x, y, 0) = (1 / vmax) * (M * lamda(2) *
max3 + (mi + M * (lamda(1) - lamda(2))) * vold(x, y, 0) - (x + y) * h)

'*********************************************************************
```

```
*********************************************
                    ElseIf z = M Then
                        max2 = 0    '(x,y,M) ((x+y=K)
                        If (s(1, 1) * vold(x - 1, y, M) + (1 - s(1, 1))
* vold(x - 1, y, M - 1) + r) >= (s(1, 2) * vold(x, y - 1, M) + (1 - s(1,
2)) * vold(x, y - 1, M - 1) + r) Then
                            max2 = s(1, 1) * vold(x - 1, y, M) + (1 -
s(1, 1)) * vold(x - 1, y, M - 1) + r
                        Else
                            max2 = s(1, 2) * vold(x, y - 1, M) + (1 -
s(1, 2)) * vold(x, y - 1, M - 1) + r
                        End If
                        vnew(x, y, M) = (1 / vmax) * (M * lamda(1) *
max2 + mi * vold(x, y, M) - h * (x + y))

'*********************************************************************
***********************************
                    Else
                        max2 = 0    '(x,y,z) (x+y=K)
                        max3 = 0
                        If (s(1, 1) * vold(x - 1, y, z) + (1 - s(1, 1))
* vold(x - 1, y, z - 1) + r) >= (s(1, 2) * vold(x, y - 1, z) + (1 - s(1,
2)) * vold(x, y - 1, z - 1) + r) Then
                            max2 = s(1, 1) * vold(x - 1, y, z) + (1 -
s(1, 1)) * vold(x - 1, y, z - 1) + r
                        Else
                            max2 = s(1, 2) * vold(x, y - 1, z) + (1 -
s(1, 2)) * vold(x, y - 1, z - 1) + r
                        End If
                        If (s(2, 1) * vold(x - 1, y, z + 1) + (1 - s(2,
1)) * vold(x - 1, y, z) + r) >= (s(2, 2) * vold(x, y - 1, z + 1) + (1 -
s(2, 2)) * vold(x, y - 1, z) + r) And (s(2, 1) * vold(x - 1, y, z + 1)
+ (1 - s(2, 1)) * vold(x - 1, y, z) + r) >= vold(x, y, z) Then
                            max3 = s(2, 1) * vold(x - 1, y, z + 1) + (1
- s(2, 1)) * vold(x - 1, y, z) + r
                        ElseIf (s(2, 2) * vold(x, y - 1, z + 1) + (1 -
s(2, 2)) * vold(x, y - 1, z) + r) >= (s(2, 1) * vold(x - 1, y, z + 1) +
(1 - s(2, 1)) * vold(x - 1, y, z) + r) And (s(2, 2) * vold(x, y - 1, z
+ 1) + (1 - s(2, 2)) * vold(x, y - 1, z) + r) >= vold(x, y, z) Then
                            max3 = s(2, 2) * vold(x, y - 1, z + 1) + (1
- s(2, 2)) * vold(x, y - 1, z) + r
                        ElseIf vold(x, y, z) >= (s(2, 1) * vold(x - 1,
y, z + 1) + (1 - s(2, 1)) * vold(x - 1, y, z) + r) And vold(x, y, z) >=
(s(2, 2) * vold(x, y - 1, z + 1) + (1 - s(2, 2)) * vold(x, y - 1, z) +
r) Then
                            max3 = vold(x, y, z)
                        End If
                        vnew(x, y, z) = (1 / vmax) * (z * lamda(1) *
max2 + (M - z) * lamda(2) * max3 + (mi + (M - z) * (lamda(1) - lamda(2)))
* vold(x, y, z) - h * (x + y))
                    End If

'*********************************************************************
***********************************
                Else
                    If z = 0 Then
                        max1 = 0  '(x,y,0)
                        max3 = 0
                        If (p(3) * (vold(x, y, 0) - c) + p(1) * vold(x
+ 1, y, 0) + p(2) * vold(x, y + 1, 0)) >= vold(x, y, 0) Then
                            max1 = p(3) * (vold(x, y, 0) - c) + p(1) *
```

```
vold(x + 1, y, 0) + p(2) * vold(x, y + 1, 0)
                        Else
                            max1 = vold(x, y, 0)
                        End If
                        If (s(2, 1) * vold(x - 1, y, 1) + (1 - s(2, 1))
* vold(x - 1, y, 0) + r) >= vold(x, y, 0) And (s(2, 1) * vold(x - 1, y,
1) + (1 - s(2, 1)) * vold(x - 1, y, 0) + r) >= (s(2, 2) * vold(x, y -
1, 1) + (1 - s(2, 2)) * vold(x, y - 1, 0) + r) Then
                            max3 = s(2, 1) * vold(x - 1, y, 1) + (1 -
s(2, 1)) * vold(x - 1, y, 0) + r
                        ElseIf (s(2, 2) * vold(x, y - 1, 1) + (1 - s(2,
2)) * vold(x, y - 1, 0) + r) >= (s(2, 1) * vold(x - 1, y, 1) + (1 - s(2,
1)) * vold(x - 1, y, 0) + r) And (s(2, 2) * vold(x, y - 1, 1) + (1 -
s(2, 2)) * vold(x, y - 1, 0) + r) >= vold(x, y, 0) Then
                            max3 = s(2, 2) * vold(x, y - 1, 1) + (1 -
s(2, 2)) * vold(x, y - 1, 0) + r
                        ElseIf vold(x, y, 0) >= (s(2, 1) * vold(x - 1,
y, 1) + (1 - s(2, 1)) * vold(x - 1, y, 0) + r) And vold(x, y, 0) >=
(s(2, 2) * vold(x, y - 1, 1) + (1 - s(2, 2)) * vold(x, y - 1, 0) + r)
Then
                            max3 = vold(x, y, 0)
                        End If
                        vnew(x, y, 0) = (1 / vmax) * (mi * max1 + M *
lamda(2) * max3 + M * (lamda(1) - lamda(2)) * vold(x, y, 0) - h * (x +
y))

'*******************************************************************
****************************************
                    ElseIf z = M Then
                        max1 = 0  '(x,y,M)
                        max2 = 0
                        If (p(3) * (vold(x, y, M) - c) + p(1) * vold(x
+ 1, y, M) + p(2) * vold(x, y + 1, M)) >= vold(x, y, M) Then
                            max1 = p(3) * (vold(x, y, M) - c) + p(1) *
vold(x + 1, y, M) + p(2) * vold(x, y + 1, M)
                        Else
                            max1 = vold(x, y, M)
                        End If
                        If (s(1, 1) * vold(x - 1, y, M) + (1 - s(1, 1))
* vold(x - 1, y, M - 1) + r) >= (s(1, 2) * vold(x, y - 1, M) + (1 - s(1,
2)) * vold(x, y - 1, M - 1) + r) Then
                            max2 = s(1, 1) * vold(x - 1, y, M) + (1 -
s(1, 1)) * vold(x - 1, y, M - 1) + r
                        Else
                            max2 = s(1, 2) * vold(x, y - 1, M) + (1 -
s(1, 2)) * vold(x, y - 1, M - 1) + r
                        End If
                        vnew(x, y, M) = (1 / vmax) * (mi * max1 + M *
lamda(1) * max2 - h * (x + y))

'*******************************************************************
****************************************
                    Else
                        max1 = 0  '(x,y,z)
                        max2 = 0
                        max3 = 0
                        If (p(3) * (vold(x, y, z) - c) + p(1) * vold(x
+ 1, y, z) + p(2) * vold(x, y + 1, z)) >= vold(x, y, z) Then
                            max1 = p(3) * (vold(x, y, z) - c) + p(1) *
vold(x + 1, y, z) + p(2) * vold(x, y + 1, z)
                        Else
```

```
                        max1 = vold(x, y, z)
                    End If
                    If (s(1, 1) * vold(x - 1, y, z) + (1 - s(1, 1))
* vold(x - 1, y, z - 1) + r) >= (s(1, 2) * vold(x, y - 1, z) + (1 - s(1,
2)) * vold(x, y - 1, z - 1) + r) Then
                        max2 = s(1, 1) * vold(x - 1, y, z) + (1 -
s(1, 1)) * vold(x - 1, y, z - 1) + r
                    Else
                        max2 = s(1, 2) * vold(x, y - 1, z) + (1 -
s(1, 2)) * vold(x, y - 1, z - 1) + r
                    End If
                    If (s(2, 1) * vold(x - 1, y, z + 1) + (1 - s(2,
1)) * vold(x - 1, y, z) + r) >= (s(2, 2) * vold(x, y - 1, z + 1) + (1 -
s(2, 2)) * vold(x, y - 1, z) + r) And (s(2, 1) * vold(x - 1, y, z + 1)
+ (1 - s(2, 1)) * vold(x - 1, y, z) + r) >= vold(x, y, z) Then
                        max3 = s(2, 1) * vold(x - 1, y, z + 1) + (1
- s(2, 1)) * vold(x - 1, y, z) + r
                    ElseIf (s(2, 2) * vold(x, y - 1, z + 1) + (1 -
s(2, 2)) * vold(x, y - 1, z) + r) >= (s(2, 1) * vold(x - 1, y, z + 1) +
(1 - s(2, 1)) * vold(x - 1, y, z) + r) And (s(2, 2) * vold(x, y - 1, z
+ 1) + (1 - s(2, 2)) * vold(x, y - 1, z) + r) >= vold(x, y, z) Then
                        max3 = s(2, 2) * vold(x, y - 1, z + 1) + (1
- s(2, 2)) * vold(x, y - 1, z) + r
                    ElseIf vold(x, y, z) >= (s(2, 1) * vold(x - 1,
y, z + 1) + (1 - s(2, 1)) * vold(x - 1, y, z) + r) And vold(x, y, z) >=
(s(2, 2) * vold(x, y - 1, z + 1) + (1 - s(2, 2)) * vold(x, y - 1, z) +
r) Then
                        max3 = vold(x, y, z)
                    End If
                    vnew(x, y, z) = (1 / vmax) * (mi * max1 + z *
lamda(1) * max2 + (M - z) * lamda(2) * max3 + (M - z) * (lamda(1) -
lamda(2)) * vold(x, y, z) - h * (x + y))

'**********************************************************************
****************************************
                End If
            End If
        End If
    Next z
  Next y
Next x

   sign = 1
   maxdif = 0
   mindif = 1000000
   For x = 0 To K Step 1
    For y = 0 To (K - x) Step 1
     For z = 0 To M Step 1
        dif = vnew(x, y, z) - vold(x, y, z)           'Diafora sto V
metaksy diadoxikwn xronikwn stigmwn
        If (dif < 0) Then
            dif = -dif
            sign = -1
        Else
            sign = 1
        End If
        If (dif < mindif) Then
            mindif = dif
        End If
        If (dif > maxdif) Then
            maxdif = dif
```

```
        End If
      Next z
     Next y
   Next x

   error = maxdif - mindif                       'Kathorismos tis apoklisis


   optavereven = maxdif * vmax                          'Ypologismos mesou
kerdous

   For x = 0 To K Step 1
    For y = 0 To (K - x) Step 1
      For z = 0 To M Step 1
        vold(x, y, z) = vnew(x, y, z)                'Antikatastasi twn
V metaksy twn xronikwn stigmwn k kai k+1, gia metabasi se epomeni xroniki
stigmi
      Next z
    Next y
   Next x

Wend                                           'telos epanalipsis

Print #filenum, "Ta dedomena tou provlimatos einai ta eksis:"
Print #filenum,
Print #filenum, "To megethos tis agoras einai M =", M
Print #filenum, "O rythmos afiksis twn taktikwn pelatwn einai lamda(1)=",
lamda(1)
Print  #filenum,  "O  rythmos  afiksis  twn  eukairiakwn  pelatwn  einai
lamda(2)=", lamda(2)
Print #filenum, "O rythmos eksypiretisis eimai mi=", mi
Print #filenum, "To kostos aporripsis enos kommatiou einai c=", c
Print #filenum, "To kostos anamonis enos pelati ana monada xronou einai
h=", h
Print #filenum, "To kerdos apo tin pwlisi enos kommatiou einai r=", r
Print #filenum, "Ta epipeda poiotitas gia ena paragomeno proion einai:",
lev
Print   #filenum,   "H  pithanotita   paragwgis   kai   oi   pithanotites
ikanopoiisis apo tous taktikous kai eukairiakous pelates antistoixa gia
ka8e epipedo poiotitas:"
For i = 1 To (lev - 1) Step 1
Print #filenum, "Epipedo poiotias ", i, ": ", p(i), ",", s(1, i), ",",
s(2, i)
Next i
Print #filenum,
Print #filenum, "To orio sfalmatos oristike sto ", merror
Print #filenum, "Petyxame sygklisi meta apo arithmo epanalipsewn:", rep
Print #filenum, "Me veltisto meso kerdos:", optavereven
Print #filenum, "sign:", sign
Print #filenum,
For z = 0 To M Step 1
Print #filenum, "O pinakas apofasewn paragwgis tha einai gia to epipedo
plithous taktikwn pelatwn ", z
    For x = 2 * M To 0 Step -1
Print #filenum, x,
        For y = 0 To (2 * M - x) Step 1     'Epilogi apofasis gia kathe
katastasi
            If (p(3) * (vold(x, y, z) - c) + p(1) * vold(x + 1, y, z) +
p(2) * vold(x, y + 1, z)) >= vold(x, y, z) Then
                dis1(x, y, z) = 1
            Else
```

```
                dis1(x, y, z) = 0
            End If
            Print #filenum, dis1(x, y, z),
        Next y
        Print #filenum, x
        Print #filenum,
    Next x
    Print #filenum, "y",
    For y = 0 To 2 * M Step 1
    Print #filenum, y,
    Next y
    Print #filenum, "y"
    Print #filenum,
Next z

Print #filenum,
For z = 1 To M Step 1
Print #filenum, "O pinakas apofasewn pwlisis gia tous taktikous pelates
tha einai gia to epipedo plithous taktikwn pelatwn ", z
    For x = 2 * M To 1 Step -1
Print #filenum, x,
        For y = 1 To (2 * M - x) Step 1        'Epilogi apofasis gia
kathe katastasi
            If (s(1, 1) * vold(x - 1, y, z) + (1 - s(1, 1)) * vold(x -
1, y, z - 1) + r) >= (s(1, 2) * vold(x, y - 1, z) + (1 - s(1, 2)) *
vold(x, y - 1, z - 1) + r) Then
                dis2(x, y, z) = 1
            Else
                dis2(x, y, z) = 2
            End If
            Print #filenum, dis2(x, y, z),
        Next y
        Print #filenum, x
        Print #filenum,
    Next x
    Print #filenum, 0,
    For y = 1 To 2 * M Step 1
        If (s(1, 2) * vold(0, y - 1, z) + (1 - s(1, 2)) * vold(0, y -
1, z - 1) + r) >= vold(0, y, z - 1) Then
            dis2(0, y, z) = 2
        Else
            dis2(0, y, z) = 0
        End If
        Print #filenum, dis2(0, y, z),
    Next y
    Print #filenum, 0
    Print #filenum,
    Print #filenum, "y",

    For y = 1 To 2 * M Step 1
    Print #filenum, y,
    Next y
    Print #filenum, "y"
    Print #filenum,
Next z

Print #filenum,
For z = 0 To (M - 1) Step 1
Print #filenum, "O pinakas apofasewn pwlisis gia tous eukairiakous
pelates tha einai gia to epipedo plithous taktikwn pelatwn ", z
    For x = 2 * M To 1 Step -1
```

```
Print #filenum, x,
        If (s(2, 1) * vold(x - 1, 0, z + 1) + (1 - s(2, 1)) * vold(x -
1, 0, z) + r) >= vold(x, 0, z) Then
            dis3(x, 0, z) = 1
        Else
            dis3(x, 0, z) = 0
        End If
        Print #filenum, dis3(x, 0, z),

        For y = 1 To (2 * M - x) Step 1           'Epilogi apofasis gia
kathe katastasi
            If (s(2, 1) * vold(x - 1, y, z + 1) + (1 - s(2, 1)) * vold(x
- 1, y, z) + r) >= (s(2, 2) * vold(x, y - 1, z + 1) + (1 - s(2, 2)) *
vold(x, y - 1, z) + r) And (s(2, 1) * vold(x - 1, y, z + 1) + (1 - s(2,
1)) * vold(x - 1, y, z) + r) >= vold(x, y, z) Then
                dis3(x, y, z) = 1
            ElseIf (s(2, 2) * vold(x, y - 1, z + 1) + (1 - s(2, 2)) *
vold(x, y - 1, z) + r) >= (s(2, 1) * vold(x - 1, y, z + 1) + (1 - s(2,
1)) * vold(x - 1, y, z) + r) And (s(2, 2) * vold(x, y - 1, z + 1) + (1
- s(2, 2)) * vold(x, y - 1, z) + r) >= vold(x, y, z) Then
                dis3(x, y, z) = 2
            ElseIf vold(x, y, z) >= (s(2, 1) * vold(x - 1, y, z + 1) +
(1 - s(2, 1)) * vold(x - 1, y, z) + r) And vold(x, y, z) >= (s(2, 2) *
vold(x, y - 1, z + 1) + (1 - s(2, 2)) * vold(x, y - 1, z) + r) Then
                dis3(x, y, z) = 0
            End If
            Print #filenum, dis3(x, y, z),
        Next y

        Print #filenum, x
        Print #filenum,
    Next x
    Print #filenum, 0,
    Print #filenum, 0,
    For y = 1 To 2 * M Step 1
        If (s(2, 2) * vold(0, y - 1, z + 1) + (1 - s(2, 2)) * vold(0, y
- 1, z) + r) >= vold(0, y, z) Then
            dis3(0, y, z) = 2
        Else
            dis3(0, y, z) = 0
        End If
        Print #filenum, dis3(0, y, z),
    Next y
    Print #filenum,
    Print #filenum, "y",
    For y = 0 To 2 * M Step 1
    Print #filenum, y,
    Next y
    Print #filenum, "y"
    Print #filenum,
Next z
End Sub
```