Αλγοριθμικά Ζητήματα Σχεδιασμού Μηχανισμών:
Ανάθεση Διεργασιών σε Μηχανές

Η
ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

Υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης
του Τμήματος Πληροφορικής
Εξεταστική Επιτροπή

από την

Αιμιλία Κοκκινέλη

ως μέρος των Υποχρεώσεων

για τη λήψη

του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ ΣΤΗΝ ΘΕΩΡΙΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Ιούλιος 2009

ΑΦΙΕΡΩΣΗ

Στους γονείς μου, Αλέξανδρο και Γεωργία, για την πολυετή και ανυπολόγιστη υποστήριξή τους.

ΕΥΧΑΡΙΣΤΙΕΣ

Αρχικά, θα ήθελα να ευχαριστήσω, τον επιβλέποντα της εργασίας μου, κύριο Σπύρο Κοντογιάννη, που με καθοδήγησε και μου αφιέρωσε πολύ από τον πολύτιμο χρόνο του, για να ολοκληρώσω την προσπάθειά μου και να πετύχω το στόχο μου.

Με την περάτωση της παρούσας μεταπτυχιακής, μου δίνεται η ευκαιρία να επισημάνω πόσο δύσκολο και κοπιαστικό είναι να ολοκληρώνεις ένα έργο που ξεκινάς, έχοντας ταυτόχρονα να διεκπεραιώσεις και άλλες υποχρεώσεις. Για μένα έγινε εφικτό χάρη στην αμέριστη κατανόηση της κας Αλίνας Χυζ και του κου Ευριπίδη Γλαβά.

Επίσης, θα ήθελα να ευχαριστήσω θερμά, την πολύ καλή φίλη, που απέκτησα κατά τη διάρκεια του ΜΠΣ, τη συμφοιτήτριά μου Αντιγόνη Γιάτσιου, για την άψογη συνεργασία που είχαμε.

Τέλος, θέλω να ευχαριστήσω ιδιαίτερα την οικογένειά μου για τη συμπαράστασή της και ελπίζω ότι τα επόμενα χρόνια θα αφιερώσω περισσότερο χρόνο στον Αλέξανδρο, την Ηλιάνα και την Εύη.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΥΡΕΤΗΡΙΟ ΣΧΗΜΑΤΩΝ

ΠΕΡΙΛΗΨΗ

Αιμιλία Κοκκινέλη του Αλεξάνδρου και της Γεωργίας. MSc, Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Ιούλιος, 2009. Αλγοριθμικά Ζητήματα Σχεδιασμού Μηχανισμών: Ανάθεση Διεργασιών σε Μηχανές. Επιβλέποντας: Σπυρίδων Κοντογιάννης.

Στην παρούσα εργασία, ασχολούμαστε με αλγοριθμικά ζητήματα σχεδιασμού μηχανισμών. Ένας μηχανισμός είναι ο προσδιορισμός των κανόνων ενός παιχνιδιού με τέτοιο τρόπο ώστε: (i) οι συμμετέχοντες στο παιχνίδι να είναι ειλικρινείς ως προς τις προθέσεις τους, και (ii) η λύση που θα προκύψει ως αποτέλεσμα του παιχνιδιού να είναι όσο το δυνατόν πλησιέστερα στην βέλτιστη λύση που θα επέλεγε ο ίδιος ο σχεδιαστής του μηχανισμού αν είχε πλήρη έλεγχο, προς όφελος ολόκληρου του συστήματος (όπως αυτό περιγράφεται μέσα από μια συνάρτηση κοινωνικής ωφέλειας).

Στην εργασία μελετάται το πρόβλημα της εύρεσης (αποδοτικών) μηχανισμών, αλλά και κάτω φραγμάτων, που σχετίζονται με το πρόβλημα της ανάθεσης διεργασιών (task scheduling) σε παράλληλες μηχανές. Στο συγκεκριμένο σενάριο, κάθε μηχανή θεωρείται ότι έχει τους δικούς της χρόνους εκτέλεσης των διεργασιών, και αυτή είναι προσωπική (κρυφή) πληροφορία κάθε μηχανής. Ο σχεδιαστής επιθυμεί να αναθέσει τις διεργασίες στις μηχανές με τέτοιο τρόπο ώστε να ελαχιστοποιείται ο χρόνος περάτωσης και της τελευταίας διεργασίας (makespan). Όμως δεν είναι σε θέση να το κάνει αυτό, γιατί δεν γνωρίζει τους πραγματικούς χρόνους εκτέλεσης των διεργασιών στις μηχανές. Ζητά λοιπόν από τις μηχανές να αποκαλύψουν τα διανύσματα των χρόνων εκτέλεσης των διεργασιών, και στη συνέχεια, με βάση αυτή την πληροφορία: (i) επιλέγει μια ανάθεση των διεργασιών στις μηχανές, και (ii) ανταμείβει κάθε μηχανή για την συνεισφορά της στην εκτέλεση των διεργασιών. Όμως, κάθε (εγωιστικά συμπεριφερόμενη) μηχανή στοχεύει στο να ελαχιστοποιήσει τη δική της προσπάθεια (χρόνο εκτέλεσης των διεργασιών που της ανατίθενται) και να μεγιστοποιήσει την ανταμοιβή που θα λάβει. Για το λόγο αυτό, ενδέχεται να δώσει εσφαλμένα δεδομένα

στο σχεδιαστή. Στόχος λοιπόν του σχεδιαστή του μηχανισμού είναι να διαλέξει τους κατάλληλους αλγόριθμους ανάθεσης διεργασιών, και καθορισμού των ανταμοιβών, ώστε: (i) οι μηχανές να μην έχουν όφελος δίνοντας ψευδή στοιχεία για τους χρόνους εκτέλεσης, και (ii) η ανάθεση που θα προκύψει να έχει makespan όσο το δυνατόν πλησιέστερο στο makespan της βέλτιστης λύσης που θα υπολόγιζε ο σχεδιαστής, λαμβάνοντας υπόψη τους πραγματικούς χρόνους εκτέλεσης. Προκειμένου ο μηχανισμός να είναι αποδοτικός, αυτοί οι δυο αλγόριθμοι θα πρέπει να είναι πολυωνυμικού χρόνου ως προς το μέγεθος του προβλήματος.

Μελετάμε το συγκεκριμένο πρόβλημα (επιλογής ενός αποδοτικού μηχανισμού για ανάθεση διεργασιών σε μηχανές) σε τρεις βασικές κατηγορίες: σε παράλληλες συσχετιζόμενες μηχανές (uniformly related machines), σε μηχανές με περιορισμό ως προς την πρόσβαση (restricted assignment machines), και σε μη συσχετιζόμενες μηχανές (unrelated machines).

EXTENDED ABSTRACT IN ENGLISH

Kokkineli, Aimilia, A. MSc, Computer Science Department, University of Ioannina, Greece. July, 2009. Algorithmic Mechanism Design: Task Scheduling Problem. Thesis Supervisor: Spyros Kontogiannis.

In the present project, we study the field of Algorithmic Mechanism Design. A mechanism defines a game in a way that: (i) everyone who participates in the game truthfully declares his/her purpose, and (ii) the outcome of the game is as close as possible to the optimum outcome that would be chosen by the mechanism designer if he had all the necessary information of the system. Mechanism designer acts for the social welfare.

We study the problem of finding efficient mechanisms, and lower bounds that connect with the task scheduling problem in parallel machines. In this scenario, each machine is supposed to need its specific time to execute a task, and this is secret (private) information for the machine. Mechanism designer wants to allocate the tasks to the machines in such a way as to minimize makespan (the maximum completion time of these machines). But, as long as he isn't aware of the real time that is needed by the machines to complete their assignment, he can't achieve such an allocation. So, he asks machines to reveal their time vectors, and then, and in accordance with this information: (i) outputs an allocation of tasks to machines, and (ii) compensate each machine for its contribution to the execution of tasks. But, every (self-interest) machine goals to minimize its own effort (execution time) and maximize the compensation is going to profit by the designer. For this reason, the machine might give false data to the mechanism designer. Mechanism designer's purpose is to choose the appropriate allocation and payment algorithms, such that: (i) machines don't benefit by declaring false information about their execution time, and (ii) the allocation outcome achieves makespan which is as close as possible to the makespan of the optimum solution that would be computed by the mechanism designer, if he was taking in to account the real

execution times. For the mechanism to be efficient, these two algorithms have to be polynomial time with respect to the length of the problem.

We study this specific problem (of selecting en efficient algorithm for the task scheduling problem) in three basic categories: uniformly related machines, restricted assignment machines, and unrelated machines.

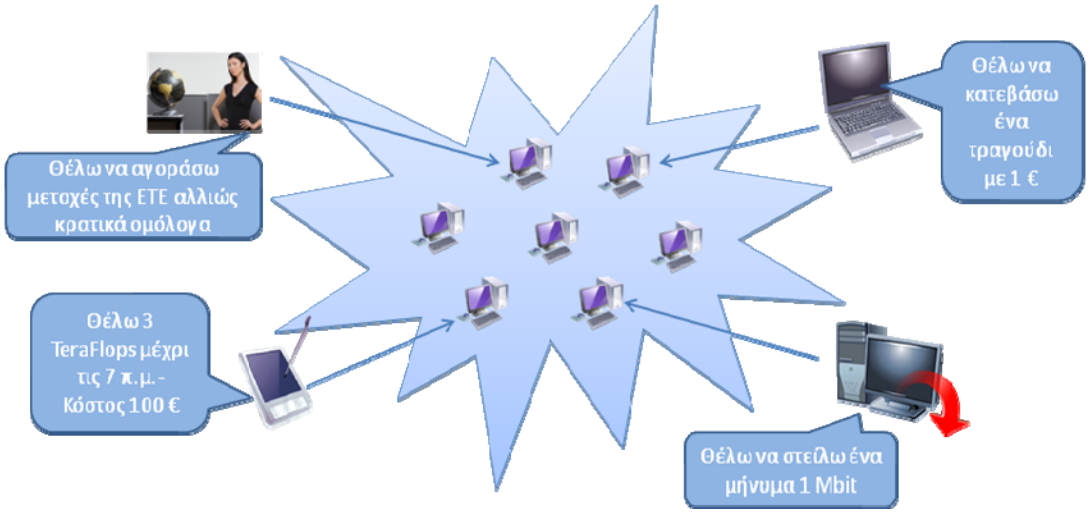
ΚΕΦΑΛΑΙΟ 1. ΕΙΣΑΓΩΓΗ

1.1. Περιγραφή Προβλήματος

Η περιοχή του **Σχεδιασμού Μηχανισμών (Mechanism Design)** ή Θεωρία Υλοποίησης (Implementation Theory) είναι υποπεριοχή της θεωρίας παιγνίων και των μικροοικονομικών και ασχολείται με την σχεδίαση πρωτοκόλλων για λελογισμένους (rational) παίκτες (βλ. παράγραφο 1.2). Γενικότερα, στο Σχεδιασμό Μηχανισμών μελετώνται προβλήματα σχετικά με τα συστήματα ψηφοφορίας (voting systems), τις πολιτικές φορολόγησης (taxation policy) και ανάθεσης δημοσίων αγαθών (allocation of public goods) κ.λπ.. Πρόσφατα ο Σχεδιασμός Μηχανισμών εφαρμόστηκε στην αλγοριθμική έρευνα με σκοπό να αντιμετωπιστούν ζητήματα όπως είναι η υπολογιστική αποδοτικότητα (computational efficiency) αλγορίθμων, η δρομολόγηση μηνυμάτων (routing of messages), η ανάθεση διεργασιών (scheduling of tasks), η κατανομή μνήμης (allocation of memory), η εύρεση συντομότερης διαδρομής (shortest path) κ.λπ..

Όλα τα παραπάνω προβλήματα έχουν κοινά χαρακτηριστικά: αποτελούνται από πολλές οντότητες (Η/Υ, μηχανές, κόμβοι, κ.λπ.) που έχουν στην κατοχή τους πόρους (resources) ή/και απαιτήσεις (requests). Επομένως, ο αλγόριθμος που θα τα επιλύσει αφενός πρέπει να λάβει υπόψη του όλες αυτές τις διαφορετικές οντότητες με όλες τις διαφορετικές προτιμήσεις τους και αφετέρου πρέπει να λειτουργεί αποδοτικά. Η ανάγκη να αντιμετωπιστούν τέτοιου είδους αλγοριθμικές προκλήσεις οδήγησε σε ένα νέο πεδίο

που ονομάζεται *Αλγοριθμικά Ζητήματα Σχεδιασμού Μηχανισμών (Algorithmic Mechanism Design)*.



Σχήμα: 1.1 Πρόβλημα ¨Ανάθεσης Πόρων στο Διαδίκτυο¨

Ένα παράδειγμα αλγοριθμικού ζητήματος σχεδιασμού μηχανισμών, είναι η δημιουργία αποδοτικών πρωτοκόλλων για το διαδίκτυο (βλ. σχήμα 1), που είναι ένα πρόβλημα που έχει παρόμοια φύση με τα προηγούμενα. Πιο συγκεκριμένα, στο διαδίκτυο υπάρχουν διαφορετικές οικονομικές οντότητες που κατέχουν, διαχειρίζονται και χρησιμοποιούν διαφορετικά μέρη του διαδικτύου. Για παράδειγμα, οι Η/Υ ανήκουν σε διαφορετικά άτομα ή οργανισμούς, οι οποίοι συμπεριφέρονται κατά κύριο λόγο προς όφελός τους. Αυτό σημαίνει ότι οι οντότητες (Η/Υ), παρουσιάζουν εγωιστική συμπεριφορά, αφού προκειμένου να επωφεληθούν από μια κατάσταση είναι δυνατό να μην ακολουθήσουν τα πρωτόκολλα ή τους αλγόριθμους. Επομένως, ένας τέτοιος αλγόριθμος ή πρωτόκολλο πρέπει να είναι εκ των προτέρων σχεδιασμένο για την αντιμετώπιση τέτοιου είδους συμπεριφορών.

Επιπλέον, η ανάπτυξη του ηλεκτρονικού εμπορίου (electronic commerce) και των ηλεκτρονικών δημοπρασιών (auctions) που διεξάγονται μέσω ιστοσελίδων, αύξησε τον ενδιαφέρον των οικονομολόγων για επίτευξη υπολογιστικής αποδοτικότητας. Ο λόγος εντοπίζεται στο γεγονός ότι αν και υπάρχουν μηχανισμοί κατάλληλοι για σύνθετους τύπους δημοπρασιών που έχουν πολλές επιθυμητές ιδιότητες, δε μπορούν να υλοποιηθούν καθώς παρουσιάζουν υπολογιστικές δυσκολίες.

Το πρόβλημα που μας απασχολεί είναι η εύρεση αποδοτικών ως προς το υπολογιστικό κόστος τους μηχανισμών (πχ, πρωτοκόλλων για τη διαχείριση και κοστολόγηση αγαθών) που αναγκάζουν τις εμπλεκόμενες, εγωιστικά φερόμενες οντότητες (ανθρώπους, οργανισμούς, προγράμματα υπολογιστών) να εκδηλώσουν με ειλικρίνεια το ενδιαφέρον τους για την αξιοποίηση αυτών των αγαθών, και ταυτόχρονα εξασφαλίζουν ότι η κατάσταση ισορροπίας στην οποία θα περιέλθει ολόκληρο το σύστημα θα είναι όσο το δυνατόν καλύτερη, με βάση κάποια συνάρτηση απόδοσης του ίδιου του συστήματος.

Πιο συγκεκριμένα, αναλύουμε μηχανισμούς που υλοποιούν το πρόβλημα ανάθεσης διεργασιών σε μηχανές (task scheduling). Μας ενδιαφέρουν οι φιλαλήθεις (truthful) μηχανισμοί, δηλαδή οι μηχανισμοί στους οποίους είναι πάντα προς όφελος των παικτών να είναι φιλαλήθεις ως προς τις πληροφορίες που αποκαλύπτουν στον μηχανισμό. Πιο συγκεκριμένα, στην παρούσα εργασία μελετάμε την ύπαρξη αποδοτικών μηχανισμών που απαρτίζονται από ένα πρωτόκολλο ανάθεσης διεργασιών και ένα σχήμα ανταμοιβών, σε παράλληλες μηχανές (πόρους). Οι μηχανές αυτές εγωιστικά επιχειρούν να μεγιστοποιήσουν το προσωπικό τους όφελος, δηλαδή, τη συνολική ανταμοιβή που θα εξασφαλίσουν μείον το υπολογιστικό κόστος που θα επωμιστούν για την εκτέλεση των διεργασιών που θα τους ανατεθούν. Από την άλλη πλευρά, ο μηχανισμός επιχειρεί (i) να εξασφαλίσει την φιλαλήθη συμμετοχή των μηχανών στη συνολική προσπάθεια για την εκτέλεση των διεργασιών, και (ii) να ελαχιστοποιήσει το μέγιστο χρόνο περάτωσης (makespan) των διεργασιών από τις μηχανές.

1.2. Δομή Εργασίας

Η μεταπτυχιακή εργασία περιέχει 6 κεφάλαια:

Στο **Κεφάλαιο 2** περιγράφονται βασικές έννοιες και ορισμοί της θεωρίας παιγνίων και το είδος των προβλημάτων που επιλύονται στα πλαίσια αυτού του κλάδου. Επίσης, περιγράφεται η ιδέα του Σχεδιασμού Μηχανισμών και δίνονται παραδείγματα συγκεκριμένων μηχανισμών. Τέλος, γίνεται η περιγραφή του Προβλήματος Ανάθεσης Διεργασιών που μελετάμε και προσδιορίζονται οι στόχοι της εργασίας.

Στο **Κεφάλαιο 3** περιγράφεται η ιδιότητα της «φιλαλήθειας » (truthfulness) και δίνονται κάποιοι τρόποι, γενικές αρχές και τεχνικές που εξασφαλίζουν την επιθυμητή ιδιότητα και οδηγούν στη δημιουργία φιλαληθών (truthful) μηχανισμών. Επιπλέον, δίνονται ορισμένοι γνωστοί στη βιβλιογραφία χαρακτηρισμοί (δηλαδή, ικανές και αναγκαίες συνθήκες) για την εξασφάλιση της φιλαλήθειας σε έναν μηχανισμό.

Στο **Κεφάλαιο 4** μελετάμε το πρόβλημα της ανάθεσης διεργασιών σε συσχετιζόμενες μηχανές, δηλαδή σε μηχανές που χαρακτηρίζονται από την ταχύτητα εξυπηρέτησής τους si. Περιγράφουμε τις απαραίτητες ιδιότητες του φιλαλήθη μηχανισμού και αναλύουμε βέλτιστους αλγόριθμους ανάθεσης διεργασιών σε μηχανές πολυωνυμικού και μη πολυωνυμικού χρόνου. Επιπλέον, δίνουμε τη μέθοδο μετατροπής ενός αλγόριθμου κλασματικών αναθέσεων σε έναν 3-προσεγγιστικό πιθανοτικό φιλαλήθη μηχανισμό.

Στο **Κεφάλαιο 5** μελετάμε μια ειδική περίπτωση του προβλήματος ανάθεσης διεργασιών σε μη συσχετιζόμενες μηχανές, όπου μια εργασία σε κάθε μηχανή απαιτεί χρόνο επεξεργασίας “low” ή “high”. Το πεδίο ορισμού είναι πολυδιάστατο. Στη συνέχεια δίνουμε μια γενική τεχνική μετατροπής οποιουδήποτε c-προσεγγιστικού αλγόριθμου σε έναν 3c-προσεγγιστικό, και φιλαλήθη κατά αναμενόμενη τιμή μηχανισμό. Μας ενδιαφέρουν οι αλγόριθμοι που ικανοποιούν την ιδιότητα της κυκλικής μονοτονικότητας που είναι ικανή και αναγκαία συνθήκη για να είναι ο μηχανισμός φιλαλήθης.

Στο **Κεφάλαιο 6** δίνεται μια μέθοδος σχεδιασμού φιλαληθών μηχανισμών για το πρόβλημα ανάθεσης διεργασιών σε μη συσχετιζόμενες μηχανές. Αυτή η μέθοδος έχει εφαρμογή και σε όλους τους ήδη γνωστούς φιλαλήθεις μηχανισμούς. Επίσης, προσδιορίζονται τα άνω και κάτω φράγματα μηχανισμών και συγκεκριμένα, προτείνεται ένας πιθανοτικός φιλαλήθης μηχανισμός για την περίπτωση ανάθεσης σε δύο μηχανές, με λόγο προσέγγισης 1,5963. Στην γενική περίπτωση, των m μηχανών, προτείνεται ένας πιθανοτικός φιλαλήθης μηχανισμός που επιτυγχάνει λόγο προσέγγισης (m+5)/2.

ΚΕΦΑΛΑΙΟ 2. ΘΕΩΡΙΑ ΠΑΙΓΝΙΩΝ-ΣΧΕΔΙΑΣΜΟΣ ΜΗΧΑΝΙΣΜΩΝ

2.1. Θεωρία Παιγνίων (Game Theory)

Η θεωρία παιγνίων είναι εκείνος ο κλάδος των εφαρμοσμένων μαθηματικών που επιχειρεί να αποδώσει με φορμαλιστικό τρόπο συμπεριφορές οντοτήτων (που συνήθως ονομάζονται παίκτες) σε ένα *περιβάλλον στρατηγικών αλληλεπιδράσεων (strategic interactions system),* όπου η επιτυχία μιας οντότητας στην επιλογή δράσης από ένα σύνολο επιλογών, εξαρτάται και από τις επιλογές των υπόλοιπων οντοτήτων που συμμετέχουν στο ίδιο περιβάλλον. Η θεωρία παιγνίων έχει ευρεία εφαρμογή σε τομείς όπως οι κοινωνικές επιστήμες, οι οικονομικές επιστήμες, η βιολογία, οι πολιτικές επιστήμες και διεθνείς σχέσεις, η φιλοσοφία, και (σχετικά πρόσφατα) η πληροφορική.

Αρχικά η θεωρία παιγνίων εστίασε το ενδιαφέρον της σε *αμιγώς ανταγωνιστικά περιβάλλοντα, όπου μια οντότητα επωφελείται πάντοτε εις βάρος των υπόλοιπων οντοτήτων*. Ένα τέτοιο παράδειγμα παιχνιδιού είναι η **«Ρίψη Νομίσματος» (Matching Pennies)** όπου δυο παίκτες συμμετέχουν στον ένα και μοναδικό γύρο του παιχνιδιού επιλέγοντας την (επάνω) όψη του δικού τους νομίσματος ο καθένας, ανεξάρτητα από το τι επιλέγει ο αντίπαλος. Και τα δυο νομίσματα των παικτών δίνονται : (i) στον πρώτο παίκτη, αν οι δυο όψεις συμφωνούν, ή (ii) στον δεύτερο παίκτη, αν οι δυο επιλεγμένες όψεις είναι διαφορετικές. Οι δυο παίκτες είναι υποχρεωμένοι εκ των προτέρων να ανακοινώσουν τη στρατηγική τους (δηλαδή, το πώς τελικά θα επιλέξουν την όψη του δικού τους νομίσματος) στον αντίπαλο, πριν πραγματικά κάνουν την επιλογή τους. Αν ονομάσουμε τους δυο παίκτες «παίκτη γραμμής» και «παίκτη στήλης» αντίστοιχα, τότε ο ακόλουθος πίνακας περιγράφει τα κέρδη των δυο παικτών, ανάλογα με το αποτέλεσμα του παιχνιδιού:

Παίκτης Γραμμής \ Παίκτης Στήλης	Κ	Γ
Κ	**1** , **-1**	**-1** , **1**
Γ	**-1** , **1**	**1** , **-1**

Σχήμα 2.1: Παιχνίδι ¨Ρίψη Νομίσματος¨

Στο συγκεκριμένο παιχνίδι είναι εύκολο να δει κανείς ότι αν ένας συγκεκριμένος παίκτης (πχ, ο παίκτης γραμμής) ανακοινώσει ντετερμινιστικά (δηλαδή, με βεβαιότητα) μια συγκεκριμένη δράση (πχ, ΚΟΡΩΝΑ) τότε ο αντίπαλός του ξέρει ακριβώς τι πρέπει να κάνει για να τον κερδίσει (πχ, ο παίκτης στήλης θα επέλεγε ακριβώς την αντίθετη δράση, δηλαδή, ΓΡΑΜΜΑΤΑ). Αυτό σημαίνει ότι και οι δυο παίκτες, εγωιστικά

σκεπτόμενοι, θα πρέπει να ανακοινώσουν στον αντίπαλο ως στρατηγική τους, όχι τη συγκεκριμένη δράση που θα υιοθετήσουν, αλλά κάποια κατανομή πιθανότητας με βάση την οποία θα επιλέξουν τη δράση τους, έτσι ώστε να κρύψουν από τον αντίπαλο την τελική τους επιλογή. Αν μελετήσει κανείς το συγκεκριμένο παιχνίδι, είναι εύκολο να διαπιστώσει ότι η μοναδική «κατάσταση ισορροπίας» του παιχνιδιού αυτού είναι κάθε παίκτης να «στρίψει δίκαια το νόμισμά του», δηλαδή να αποφασίσει τη δράση του ανεξάρτητα από το τι κάνει ο αντίπαλος, χρησιμοποιώντας την ομοιόμορφη κατανομή. Η συγκεκριμένη στρατηγική εξασφαλίζει για καθέναν από τους παίκτες μηδενικό (αναμενόμενο) κέρδος, ενώ για οποιαδήποτε άλλη στρατηγική του ενός παίκτη (κατανομή πιθανότητας για την επιλογή από το {ΚΟΡΩΝΑ,ΓΡΑΜΜΑΤΑ}, υπάρχει η κατάλληλη στρατηγική του αντιπάλου που εξασφαλίζει θετικό κέρδος για τον αντίπαλο και αρνητικό κέρδος (δηλαδή, ζημιά) για τον ίδιο τον παίκτη.

Η θεωρία παιγνίων γρήγορα εξαπλώθηκε και σε πιο γενικά σενάρια αλληλεπίδρασης, όπου *κάθε οντότητα έχει τις δικές της προτιμήσεις (όχι απαραίτητα πάντα σε σύγκρουση με αυτές των υπόλοιπων οντοτήτων)* για κάθε δυνατή κατάσταση στην οποία μπορεί να βρεθεί ολόκληρο το σύστημα. Ως παράδειγμα αναφέρουμε το πολύ γνωστό παιχνίδι **«Δίλημμα του Φυλακισμένου» (The Prisoner's Dilemma)**: Δυο κρατούμενοι έχουν συλληφθεί για τη διάπραξη μιας τροχαίας παράβασης (που επισύρει ποινή φυλάκισης 1 χρόνου), όμως η αστυνομία τους υποψιάζεται για τη διάπραξη ενός πολύ σοβαρότερου αδικήματος (π.χ. κλοπή) που επισύρει ποινή φυλάκισης 10 χρόνων. Όμως, δεν υπάρχουν επαρκή στοιχεία για την απόδειξη της ενοχής τους γι' αυτό το σοβαρό αδίκημα, έτσι λοιπόν η αστυνομία επιχειρεί να τους αποσπάσει μια ομολογία ενοχής, κάνοντας το εξής: Οι δυο κρατούμενοι τοποθετούνται σε δυο χωριστά κελιά, δίχως δυνατότητα επικοινωνίας, και η αστυνομία ανακοινώνει και στους δυο ότι, ανάλογα με το αν θα προδώσουν (Π) το συνεργάτη τους ή θα σιωπήσουν (Σ), θα εκτίσουν ποινή φυλάκισης όπως περιγράφεται στον ακόλουθο πίνακα:

Παίκτης Γραμμής \ Παίκτης Στήλης	Π	Σ
Π	**-5 , -5**	**0 , -10**
Σ	**-10 , 0**	**-1 , -1**

Σχήμα 2.2 Παιχνίδι ¨Δίλλημα του Φυλακισμένου¨

Παρατηρούμε ότι αν ο παίκτης γραμμής προδώσει, τότε η ποινή του δεν είναι βέβαιη αλλά εξαρτάται και από το τι θα κάνει ο παίκτης στήλης (αν και οι δυο προδώσουν τότε μοιράζονται την ποινή των 10 χρόνων, αν όμως ο παίκτης στήλης σιωπήσει, τότε ο παίκτης γραμμής αφήνεται ελεύθερος ενώ ο παίκτης στήλης εκτίει ολόκληρη την ποινή). Στο συγκεκριμένο παιχνίδι υπάρχει μια προφανής λύση που θα ήταν προς όφελος και των δυο παικτών: Αν και οι δυο σιωπούσαν, τότε η αστυνομία θα μπορούσε να τους καταδικάσει μόνο για την τροχαία παράβαση (λόγω έλλειψης στοιχείων για το σοβαρότερο αδίκημα) και καθένας τους θα έπρεπε να εκτίσει ποινή φυλάκισης ενός χρόνου. Παρόλα αυτά, εύκολα βλέπει κανείς ότι (πχ) ο παίκτης γραμμής έχει συμφέρον να προδώσει, ανεξάρτητα από το τι θα κάνει ο παίκτης στήλης, αφού και στις δυο περιπτώσεις επιλογής δράσης για τον παίκτης στήλης, η ποινή για τον παίκτη γραμμής είναι μικρότερη όταν προδίδει. Έτσι, καταλήγουμε ότι (δεδομένου ότι οι δυο παίκτες δεν επικοινωνούν μεταξύ τους, και άρα δε μπορούν να συνάψουν κάποιου είδους συμφωνία) και οι δυο ύποπτοι θα προδώσουν ο ένας τον άλλον και θα εκτίσουν ποινή φυλάκισης 5 χρόνων ο καθένας.

Επιπλέον, υπάρχουν παιχνίδια όπου *οι παίκτες θέλουν να συντονίσουν τη συμπεριφορά τους, αλλά έχουν αντικρουόμενες επιθυμίες*. Ένα τέτοιο παράδειγμα ονομάζεται **«BoS» (Bach or Stravinsky)**: Δύο άτομα θέλουν να πάνε μαζί σε ένα κονσέρτο μουσικής του Bach ή του Stravinsky. Αν και το κύριο μέλημά τους είναι να πάνε μαζί, ο ένας επιθυμεί

περισσότερο τον Bach και ο άλλος τον Stravinsky. Το παράδειγμα απεικονίζεται με τον παρακάτω πίνακα:

Παίκτης Γραμμής \ Παίκτης Στήλης	B	S
B	**2** , **1**	**0** , **0**
S	**0** , **0**	**1** , **2**

Σχήμα 2.3 Παιχνίδι ¨BoS¨

Παρατηρούμε ότι στο συγκεκριμένο παιχνίδι υπάρχουν δύο σημεία ισορροπίας που θα ικανοποιούσαν την ανάγκη των δύο παικτών να συντονίσουν τη συμπεριφορά τους και να πάνε μαζί στο κονσέρτο μουσικής. Επομένως, υπάρχουν τρεις στρατηγικές για το παιχνίδι, δύο αμιγείς και μια μικτή. Στις αμιγείς στρατηγικές οι παίκτες: (i) επιλέγουν και οι δύο Bach ή (ii) επιλέγουν και οι δύο Stravinsky. Στη μικτή στρατηγική ο παίκτης επιλέγει τη δράση του με βάση κάποια κατανομή πιθανότητας. Ο παίκτης κάνει την κατανομή πιθανότητας με τέτοιο τρόπο ώστε να του εξασφαλίζεται το ίδιο αναμενόμενο όφελος είτε αποφασισθεί τελικά να πάνε Bach είτε Stravinsky. Το μικτό προφίλ στρατηγικών είναι όταν ο παίκτης γραμμής με πιθανότητα 1/3 επιλέγει Bach και με πιθανότητα 2/3 επιλέγει Stravinsky, ενώ ο παίκτης στήλης με πιθανότητα 1/3 επιλέγει Stravinsky και με πιθανότητα 2/3 Bach.

2.1.1. Ιστορία της Θεωρίας Παιγνίων

Αν και υπήρχαν ορισμένες εξελίξεις νωρίτερα, η θεωρία παιγνίων καθιερώθηκε με (και πήρε το όνομά της από) το βιβλίο *Theory of Games and Economic Behavior* [John von

Neumann, Oskar Morgenstern (1944)]. Γνώρισε δε ιδιαίτερη ανάπτυξη κατά τη δεκαετία του 1950, όταν υιοθετήθηκε ευρέως ως θεμελιώδες αναλυτικό εργαλείο από τις οικονομικές επιστήμες. Αργότερα (δεκαετία του 1970) εφαρμόσθηκε στη βιολογία, ιδιαίτερα στον τομέα της που σήμερα είναι γνωστός ως Εξελικτική Θεωρία Παιγνίων. Τα τελευταία 15 χρόνια γνωρίζει ιδιαίτερη επιτυχία και αμέτρητες εφαρμογές στην Πληροφορική, με έμφαση σε αλγοριθμικά ζητήματα που σχετίζονται κυρίως με τρόπους κατασκευής ισορροπιών, δημιουργία αποδοτικών μηχανισμών για την επιβολή μιας επιθυμητής (για το σύστημα) ισορροπίας, ταχύτητα σύγκλισης ακολουθιών βημάτων εγωιστικών επιλογών από τους παίκτες σε κάποια ισορροπία, κ.λπ.

Σήμερα η θεωρία παιγνίων αναγνωρίζεται ευρέως ως ένα από τα πιο πετυχημένα παρακλάδια των εφαρμοσμένων μαθηματικών. Για παράδειγμα, ήδη έχουν απονεμηθεί οκτώ βραβεία Nobel σε επιστήμονες για τα επιτεύγματά τους στη θεωρία παιγνίων.

2.1.2. Ορολογία Θεωρίας Παιγνίων

Η θεωρία Παιγνίων είναι ένα σύνολο από ιδέες και τεχνικές που χρησιμοποιούνται στην ανάλυση μαθηματικών μοντέλων αντικρουόμενων συμφερόντων. Δεν απαντάει στην ερώτηση ¨πώς να παίξεις ένα παιχνίδι¨, αλλά περιγράφει τις ιδιότητες μιας συγκεκριμένης στρατηγικής επιλογής, αυτής που ο παίκτης επιθυμεί περισσότερο. Ακόμα κι αν τελικά η ανάλυση προτείνει την καλύτερη δυνατή στρατηγική για έναν παίκτη, το κάνει υποθέτοντας ότι όλοι οι υπόλοιποι παίκτες έχουν επιλέξει με το καλύτερο δυνατό τρόπο για τους ίδιους.

Παρακάτω θα περιγράψουμε τα κοινά χαρακτηριστικά που έχουν τα παραπάνω παραδείγματα, και γενικότερα όλα τα παιχνίδια:

1. Υπάρχουν πάντα τουλάχιστον δύο συμμετέχοντες (participants) N, $|N| \geq 2$, που περιγράφονται ως οντότητες ή πράκτορες (agents), ή παίκτες (players) κ.λπ. Στο εξής θα ονομάζονται ***παίκτες.***

2. Κάθε παιχνίδι αποτελείται από μια ένα σύνολο (πιθανών) δράσεων που είναι είτε αποφάσεις των παικτών (π.χ. επιλογή κορώνα ή γράμματα), είτε αποτελέσματα τυχαίων γεγονότων (π.χ. αποτέλεσμα ρίψης νομίσματος). Επομένως, κάθε παίκτης πρέπει να πάρει αποφάσεις κατά τη διάρκεια του παιχνιδιού. Δηλαδή, $\forall i \in N$, S_i είναι το σύνολο δράσεων του συγκεκριμένου παίκτη ($|S_n| \geq 2$). Η **στρατηγική[1] (strategy)** ενός παίκτη είναι ο αλγόριθμος με βάση τον οποίο ο συγκεκριμένος παίκτης (ανακοινώνει ότι) θα επιλέξει τη δική του δράση από το S_n, ανεξάρτητα από τους άλλους παίκτες.

3. Ο παίκτης έχει μια **συνάρτηση ωφέλειας (utility)** ως προς το αποτέλεσμα του παιχνιδιού, που αντανακλά τις προτιμήσεις του συγκεκριμένου παίκτη επί όλων των πιθανών αποτελεσμάτων του παιχνιδιού: $\forall i \in N$, $U_i : \underset{q \in N}{\times} S_q \to R$ είναι η συνάρτηση ωφέλειας του παίκτη n, που υποδηλώνει (μέσω ενός πραγματικού αριθμού για κάθε προφίλ στρατηγικών για όλους τους παίκτες) τη σημαντικότητα του αποτελέσματος του παιχνιδιού για το συγκεκριμένο προφίλ στρατηγικών.

4. Στο τέλος κάθε παιχνιδιού, κάθε παίκτης λαμβάνει μια **αποζημίωση** ή καταβάλλει μια **πληρωμή (payoff)**, η οποία υποθέτουμε ότι είναι πραγματικός αριθμός. Στο παράδειγμα του παιχνιδιού «ρίψη νομίσματος» ο παίκτης γραμμής

[1] Για παράδειγμα, ένας παίκτης μπορεί να αποφασίσει (και να ανακοινώσει) ότι με βεβαιότητα θα υιοθετήσει μιας συγκεκριμένη δράση (οπότε μιλάμε για ΑΜΙΓΗ στρατηγική) ή μπορεί να επιλέξει τη δράση του ως αποτέλεσμα ενός τυχαίου και ανεξάρτητου πειράματος που θα εκτελέσει πάνω στο σύνολο δράσεών του, S_n, με βάση κάποια κατανομή πιθανότητας (οπότε μιλάμε για ΜΙΚΤΗ στρατηγική, που περιγράφεται πλήρως από την κατανομή πιθανότητας).

ζημιώνεται 1 ευρώ ή κερδίζει 1 ευρώ, ανάλογα με το αν τα δυο νομίσματα έχουν τελικά διαφορετική ή την ίδια όψη.

2.2. Λελογισμένη Συμπεριφορά (Rational Behavior)

Μια θεμελιώδης υπόθεση της θεωρίας παιγνίων (τουλάχιστον για τα παιχνίδια πλήρους γνώσης που μελετάμε στην παρούσα εργασία) είναι ότι καθένας από τους παίκτες επιδεικνύει μια «λελογισμένη συμπεριφορά», δηλαδή, επιλέγει τη στρατηγική του με τέτοιο τρόπο ώστε, ανάλογα με τις στρατηγικές των υπόλοιπων παικτών, να μεγιστοποιεί την τιμή της προσωπικής του συνάρτησης ωφέλειας.

Πιο συγκεκριμένα, αν θεωρήσουμε το σύνολο $\Delta_n = \{x \in [0,1]^{|Sk|}: \Sigma_{s \in Sk}\ x(s) = 1\}$ όλων των μικτών στρατηγικών που μπορεί να υιοθετήσει οποιοσδήποτε παίκτης $k \in N$, τότε η υπόθεση της λελογισμένης συμπεριφοράς μας εξασφαλίζει πως για επιλογή στρατηγικών των άλλων παικτών, $\sigma_k \in \Delta_k$, $k \neq n$, ο παίκτης n επιλέγει μια δική του στρατηγική $\sigma_n \in \Delta_n$ που αναθέτει θετική μάζα πιθανότητας μόνο σε δράσεις που μεγιστοποιούν το προσωπικό του όφελος:

$$\forall s \in S_n,\ \sigma_n(s) > 0 \rightarrow s \in \arg\max\{ U_n(s_n,\sigma_{-n}) : s_n \in S_n\}$$

όπου $U_n(s_n,\sigma_{-n}) = \mathbf{E}_{s_{-n} \sim \sigma_{-n}}\{ U_n(s_n , s_{-n}) \}$ είναι η αναμενόμενη ωφέλεια που θα έχει ο παίκτης n διαλέγοντας τη δράση s_n, όταν οι υπόλοιποι παίκτες διαλέγουν τις δράσεις τους σύμφωνα με το προφίλ στρατηγικών $\sigma_{-n} = (\sigma_k)_{k \neq n}$.

2.3. Λύσεις Παιχνιδιών –Κατάσταση Ισορροπίας (Equilibrium)

Οι παραδοσιακές εφαρμογές της θεωρίας παιγνίων, κατά ένα μεγάλο ποσοστό, λαμβάνουν ως πιθανές λύσεις των παιχνιδιών, κάποιας μορφής **κατάσταση ισορροπίας**

στην οποία μπορεί να περιέλθει ολόκληρο το σύστημα, ως αποτέλεσμα της «λελογισμένης συμπεριφοράς» των οντοτήτων που περιλαμβάνει. Για παράδειγμα, ένα διάνυσμα επιλογών δράσης για όλους τους παίκτες οδηγεί σε **ισορροπία Nash**, αν οποιοσδήποτε παίκτης δε μπορεί να βρει μια εναλλακτική δράση που να του εξασφαλίζει μια κατάσταση που είναι γι' αυτόν *αυστηρά πιο προτιμητέα* από την τρέχουσα κατάσταση, δεδομένου ότι οι υπόλοιποι παίκτες διατηρούν αναλλοίωτες τις δικές τους επιλογές (αυτό καλείται *μονομερής αλλαγή δράσης* από το συγκεκριμένο παίκτη).

$$\text{Για κάθε παίκτη } n \in N, \text{ έχουμε } U_n(s_n,\sigma_{-n}) \geq U_n(s'_n,\sigma_{-n}),\ \forall s \in S_n$$

Στο παράδειγμα της Ρίψης Νομίσματος, είναι σχετικά απλό να δείξει κανείς ότι η μοναδική ισορροπία Nash, είναι και για τους δυο παίκτες να επιλέξουν (και να ανακοινώσουν αυτή την επιλογή τους) να «στρίψουν το νόμισμα» με δίκαιο τρόπο, ώστε η πιθανότητα επιλογής Κορώνας ή Γραμμάτων να είναι ακριβώς η ίδια. Όσο για το Δίλημμα του Φυλακισμένου, η μοναδική ισορροπία Nash επιβάλλει και στους δυο παίκτες να προδώσουν ο ένας τον άλλο, γιατί οτιδήποτε επιλέξει ο αντίπαλος, κάθε φυλακισμένος έχει ξεκάθαρο όφελος να προδώσει παρά να σιωπήσει. Τέλος, στο παράδειγμα Bach or Stravinsky έχουμε δύο ισορροπίες Nash: (Bach, Bach) και (Stravinsky, Stravinsky). Οι δύο αμιγείς στρατηγικές είναι: (i) όταν και οι δύο παίκτες επιλέγουν πάντα Bach και (ii) όταν και οι δύο παίκτες επιλέγουν πάντα Stravinsky. Το μικτό προφίλ στρατηγικών είναι όταν ο παίκτης γραμμής με πιθανότητα 2/3 επιλέγει Bach και με πιθανότητα 1/3 επιλέγει Stravinsky, ενώ ο παίκτης στήλης με πιθανότητα 1/3 επιλέγει Stravinsky και με πιθανότητα 2/3 Bach.

Φυσικά, η ισορροπία Nash δεν είναι η μοναδική κατάσταση ισορροπίας που θα μπορούσε να θεωρηθεί ως λύση ενός παιχνιδιού. Ανάλογα με το πεδίο εφαρμογής της Θεωρίας Παιγνίων, έχουν κατά καιρούς οριστεί πάρα πολλά **σκεπτικά λύσεων** (solution concepts). Στην πραγματικότητα, η ισορροπία Nash έχει δεχθεί σφοδρότατη κριτική για το γεγονός ότι αφορά αποκλειστικά τα σενάρια μονομερούς αλλαγής δράσης από τους

παίκτες, ενώ στην πραγματικότητα οι οντότητες μπορεί να συνεργάζονται για ένα μακροπρόθεσμο στόχο, να συνάπτουν κρυφές ή φανερές συμφωνίες για συντονισμένη συμπεριφορά, να προσποιούνται αλλαγή δράσης προκειμένου να παρασύρουν τους υπόλοιπους παίκτες, κ.λπ. Η αλήθεια είναι όμως ότι μεταξύ των σκεπτικών λύσεων που έχουν κατά καιρούς προταθεί, η ισορροπία Nash είναι το *μοναδικό σκεπτικό λύσης* που (μέχρι σήμερα) γνωρίζουμε ότι έχει μια σημαντική ιδιότητα: *Υπάρχει μια τέτοια λύση, για κάθε παιχνίδι με πεπερασμένο πλήθος παικτών και πεπερασμένο πλήθος επιλογών ανά παίκτη.*

2.4. Σχεδιασμός Μηχανισμών (Mechanism Design)

Ο *Σχεδιασμός Μηχανισμών*[2] είναι κλάδος της θεωρίας παιγνίων και των μικροοικονομικών και ασχολείται με την σχεδίαση πρωτοκόλλων για λελογισμένους παίκτες, οι οποίοι επιλέγουν πάντα εκείνη την στρατηγική που θα μεγιστοποιήσει το *προσωπικό τους όφελος (self-interest)*, δηλαδή την καλύτερη στρατηγική για το παιχνίδι που συμμετέχουν. Ο Σχεδιασμός Μηχανισμών αφορά την κατασκευή μηχανισμών, δηλαδή, τον σχεδιασμό των κανόνων, των πολιτικών και των περιορισμών ενός παιχνιδιού (συστήματος), με στόχο την επίτευξη ενός συγκεκριμένου επιθυμητού αποτελέσματος [NiRo99].

Από τα παρακάτω παιχνίδια φαίνεται ότι οι κανόνες, οι πολιτικές και οι περιορισμοί που θα τεθούν θα επηρεάσουν το αποτέλεσμα του παιχνιδιού:

- ***Δημοπρασία (Auction):*** το αποτέλεσμα μιας δημοπρασίας εξαρτάται από το αν οι προσφορές (bids) υποβάλλονται με σφραγισμένους φακέλους ή προφορικά (public auction). Οι παίκτες είναι σημαντικό να γνωρίζουν αν η προσφορά που θα υποβάλουν για ένα αγαθό θα παραμείνει απόρρητη ή θα γίνει γνωστή στους

[2] Ο όρος *Σχεδιασμός Μηχανισμών* (Mechanism Design) είναι επίσης γνωστός και ως Θεωρία Εφαρμογών (Implementation Theory).

άλλους παίκτες, καθώς αυτός είναι ένας παράγοντας που θα επηρέαζε το μέγεθος της τελικά προσφερόμενης τιμής και άρα θα επηρέαζε το αποτέλεσμα του παιχνιδιού.

- ***Ανάθεση Διεργασιών (Job Scheduling):*** οι πολιτικές ανάθεσης που θα τεθούν επηρεάζουν τον τρόπο και το χρόνο που θα γίνει η ανάθεση των διεργασιών στις μηχανές. Για παράδειγμα, είναι δυνατό να τεθούν προτεραιότητες στην εκτέλεση των διεργασιών, περιορισμοί στις μηχανές κ.λπ.

- ***Δημόσιο Έργο (Public Project):*** ο τρόπος με τον οποίο διαμοιράζεται το κόστος του δημοσίου έργου στο κοινωνικό σύνολο (π.χ. μέσω πολιτικών φορολόγησης) επηρεάζει την απόφαση για την ανάληψη ή όχι του έργου. Σε περίπτωση που το δημόσιο όφελος που θα προκύψει από την κατασκευή του δημοσίου έργου δεν υπερβαίνει το κόστος του, το έργο δε θα αναληφθεί.

2.4.1. Ορισμός Προβλήματος Σχεδιασμού Μηχανισμών

Ένας μηχανισμός (f , P) σε γενικές γραμμές αποτελείται από δύο μέρη: τη **συνάρτηση αποτελέσματος** (outcome function), που καλείται και **αλγόριθμος ανάθεσης** (allocation rule), και από τον **αλγόριθμο κοστολόγησης** (payment algorithm). Η συνάρτηση εξόδου καθορίζει την τελική κατάσταση ολόκληρου του συστήματος, ενώ ο αλγόριθμος κοστολόγησης τις ανταμοιβές ή χρεώσεις (ανάλογα με το πρόσημο) των συμμετεχόντων στο σύστημα. Πιο συγκεκριμένα, ένας μηχανισμός (f , P) περιγράφεται ως εξής[NiRo99]:

- A: Το σύνολο **ενδεχομένων καταστάσεων (alternatives)** στις οποίες μπορεί να περιέλθει το σύστημα.

- N = { 1 , 2 , … , n } : Ένα σύνολο λελογισμένων παικτών.

- $\forall i \in N$, V_i είναι το σύνολο των **συναρτήσεων αποτίμησης** (valuation functions) των ενδεχομένων καταστάσεων από το A. Η συνάρτηση αποτίμησης, ουσιαστικά αποτελεί μια ποσοτικοποίηση της αξίας που αποκομίζει ο παίκτης από την εκάστοτε κατάσταση στην οποία μπορεί να περιέλθει το σύστημα, θεωρώντας μια ενιαία νομισματική μονάδα. Το καρτεσιανό γινόμενο $V = X_{i \in N}$, V_i είναι ο χώρος συναρτήσεων αποτίμησης για όλους τους παίκτες. Η συνάρτηση αποτίμησης αποτελεί κρυφή (private) πληροφορία για τον παίκτη.

- $f : V_1 \times \ldots \times V_n \rightarrow R$ είναι η συνάρτηση αποτελέσματος (outcome function) ή κανόνας ανάθεσης (allocation rule), που δέχεται ως είσοδο τις αποτιμήσεις *που δηλώνουν* οι παίκτες, και επιστρέφει την τελική κατάσταση του συστήματος. Δηλαδή, για κάθε διάνυσμα δηλώσεων αποτίμησης, $\nu \in V$, $f(\nu) \in A$ είναι η κατάσταση που επιλέγει ο μηχανισμός ως αποτέλεσμα του παιχνιδιού.

- $\forall i \in N$, $P_i : V_1 \times \ldots \times V_n \rightarrow A$ είναι η συνάρτηση κοστολόγησης (payment function) για τον παίκτη i, σε σχέση με τις δηλώσεις συναρτήσεων αποτίμησης που κάνουν όλοι οι παίκτες. Δηλαδή, $\forall \nu \in V$, $P_i(\nu)$ είναι η τιμή που είτε καλείται να πληρώσει (όταν $P_i(\nu) > 0$), ή με την οποία αποζημιώνεται (όταν $P_i(\nu) < 0$) ο παίκτης i, για το συγκεκριμένο διάνυσμα δηλώσεων συναρτήσεων αποτίμησης.

- $\forall i \in N$, $U_i : V \times V_i \rightarrow R$ είναι η συνάρτηση ωφέλειας που ενδιαφέρεται ο παίκτης i να βελτιστοποιήσει. Η U_i δέχεται ως είσοδο (i) το προφίλ των δηλώσεων συναρτήσεων αποτίμησης $b \in V$ όλων των παικτών (του παίκτη i συμπεριλαμβανομένου), και (ii) την πραγματική συνάρτηση αποτίμησης $v_i \in V_i$ του παίκτη i, και επιστρέφει τον πραγματικό αριθμό $U_i(\mathbf{b} ; v_i)$ που ορίζεται ως εξής:

$\forall \mathbf{b_{-i}} \in V_{-i}, \forall v_i, b_i \in V_i,\ U_i(b_i, \mathbf{b_{-i}} ; \nu) = \nu_i(f(b_i, \mathbf{b_{-i}})) - P_i(b_i, \mathbf{b_{-i}})$	Εξ. 2.1

Δηλαδή, αν ο παίκτης i δηλώσει (ενδεχομένως ψευδώς) μια συνάρτηση αποτίμησης b_i, ενώ η πραγματική του συνάρτηση αποτίμησης είναι η v_i, και οι άλλοι παίκτες δηλώσουν (ψευδώς ή αληθώς) το προφίλ συναρτήσεων αποτίμησης $b_{-i} = (b_k)_{k \neq i}$, τότε η ωφέλεια του παίκτη i είναι ακριβώς: $U_i(b_i, b_{-i} ; v_i) = v_i(f(b_i , b_{-i})) – P_i(b_i , b_{-i})$. Όμως, η αποτίμηση της λύσης $f(b_i , b_{-i})$ που προτείνει ο μηχανισμός (σε σχέση με τις δηλώσεις των παικτών) γίνεται ως προς την *πραγματική* συνάρτηση αποτίμησης του παίκτη i, και όχι την δηλωθείσα. Εντούτοις, τόσο η προτεινόμενη λύση $f(b_i , b_{-i})$, όσο και η κοστολόγηση $P_i(b_i , b_{-i})$, εξαρτώνται από τις *δηλώσεις* αποτιμήσεων (και όχι τις πραγματικές αποτιμήσεις) των παικτών, αφού αυτές πρέπει να καθοριστούν από το μηχανισμό που δεν μπορεί να γνωρίζει παρά μόνο τις δηλώσεις των παικτών.

Ο μηχανισμός θεωρείται λοιπόν ότι εκτελεί το εξής πρωτόκολλο:

- **ΒΗΜΑ 1**: Κάθε παίκτης πρέπει να δηλώσει την προσωπική του συνάρτηση αποτίμησης $v_i' \in V_i$.

- **ΒΗΜΑ 2:** Ο μηχανισμός καθορίζει ως τελική κατάσταση του συστήματος την κατάσταση $f(\mathbf{v}') \in A$.

- **ΒΗΜΑ 3:** Ο μηχανισμός χρεώνει (ή επιβραβεύει, ανάλογα με το πρόσημο) κάθε παίκτη $i \in N$ με το κόστος (ή κέρδος) $P_i(v') \in R$.

- **ΒΗΜΑ 4:** Η ωφέλεια κάθε παίκτη $i \in N$, με πραγματική συνάρτηση αποτίμησης $v_i \in V_i$, είναι η εξής: $U_i(v_i', v_{-i}' ; v) = v_i(f(v_i', v_{-i}')) – P_i(v_i', v_{-i}')$.

2.4.2. Ο Στόχος του Μηχανισμού και των Παικτών

Όπως έχει ήδη αναφερθεί, στόχος των λελογισμένων παικτών είναι να μεγιστοποιήσουν τις τιμές των προσωπικών συναρτήσεων ωφέλειάς τους. Αυτό σημαίνει ότι αν ένας μηχανισμός (f , P) ζητήσει από τους παίκτες να δηλώσουν την πραγματική τους

συνάρτηση αποτίμησης $v_i \in V_i$, αυτοί είναι δυνατό να δηλώσουν κάποια άλλη ψευδή συνάρτηση αποτίμησης $b_i \in V_i$, προκειμένου να επιτύχουν κάτι τέτοιο. Θα πρέπει λοιπόν ο ίδιος ο μηχανισμός να εξασφαλίζει ότι δε μπορεί να συμβεί κάτι τέτοιο (δηλαδή, ότι κάποιος παίκτης μπορεί να κερδίσει κάνοντας ψεύτικη δήλωση).

Επιπλέον, ο σχεδιαστής ενός μηχανισμού έχει ως στόχο του την κατασκευή του μηχανισμού (f , P) έτσι ώστε η τελικά επιλεγμένη κατάσταση f(b) για το προφίλ δηλώσεων b των παικτών, να έχει αξία όσο το δυνατόν πλησιέστερη ως προς την κατάσταση που θα έπρεπε να επιλέξει, αν ήξερε το προφίλ v των πραγματικών συναρτήσεων αποτίμησης των παικτών. Πιο συγκεκριμένα, υπάρχει μια ***συνάρτηση κοινωνικής επιλογής (social choice function)*** $g : V_1 \times \ldots \times V_n \rightarrow A$, που καθορίζει την κατάσταση του συστήματος ως συνάρτηση των πραγματικών δηλώσεων αποτίμησης.

Για παράδειγμα, η συνάρτηση

$$\boldsymbol{\alpha^* \in \arg\max_{a \in A} \{ g_{sum}(v) = \Sigma_{i \in N}[\, v_i(a) - P_i(v) \,] \}}$$

επιστρέφει μια κατάσταση **α*** του συστήματος που μεγιστοποιεί τη συνολική ωφέλεια των παικτών, όταν το προφίλ των πραγματικών συναρτήσεων αποτίμησης είναι το **v**.

Αντίθετα, η συνάρτηση κοινωνικής επιλογής

$$\boldsymbol{\alpha^* \in \arg\max_{a \in A} \{ g_{min}(v) = \min_{i \in N}[\, v_i(a) - P_i(v) \,] \}}$$

επιλέγει μια κατάσταση **α*** που μεγιστοποιεί την ωφέλεια του πιο δυσαρεστημένου παίκτη, δεδομένου και πάλι ότι το προφίλ των πραγματικών συναρτήσεων αποτίμησης είναι το **v**.

Στόχος λοιπόν του σχεδιαστή μηχανισμών είναι η επιλογή (από πλευράς του μηχανισμού) εκείνης της τελικής κατάστασης του συστήματος $f(\mathbf{b}) \in A$ της οποίας η αξία να είναι όσο το δυνατόν πιο κοντά στην αξία της κατάστασης **α*** που θα επέλεγε αν ήξερε εξ αρχής τις πραγματικές συναρτήσεις προτίμησης των παικτών.

Συνοψίζοντας, ο μηχανισμός πρέπει:

1. Να παροτρύνει τους παίκτες να πουν την αλήθεια, πράγμα που μπορεί να καταφέρει αν οι παίκτες αποδεδειγμένα γνωρίζουν ότι δε μπορούν να κερδίσουν λέγοντας ψέματα.

2. Να καταφέρει κάτι τέτοιο με όσο το δυνατόν μικρότερη απώλεια, ως προς την ποιότητα της λύσης που παρέχει η συνάρτηση κοινωνικής επιλογής g (π.χ. η λύση που επιλέγεται από την g, να είναι το πολύ δ φορές χειρότερη από τη βέλτιστη λύση που θα μπορούσε να προταθεί, αν εξ αρχής ήταν γνωστές οι πραγματικές αποτιμήσεις των παικτών και είχαμε απεριόριστη υπολογιστική ισχύ).

Ο πρώτος (και βασικότερος) στόχος ενός μηχανισμού (f, P), να εξασφαλίσει ότι για καθέναν από τους παίκτες αποτελεί ***κυρίαρχη στρατηγική***[3] ***(dominant strategy)*** η δήλωση των πραγματικών τους συναρτήσεων αποτίμησης, ονομάζεται *φιλαλήθεια* (truthfulness). Πιο συγκεκριμένα:

ΟΡΙΣΜΟΣ 2.1 [Φιλαλήθεια] Ένας μηχανισμός (f , P) είναι φιλαλήθης αν και μόνο αν για κάθε παίκτη i, για κάθε προφίλ (ειλικρινών ή μη) δηλώσεων συναρτήσεων αποτίμησης $\mathbf{v_{-i}} \in V_{-i}$ των άλλων παικτών, και για κάθε ζευγάρι πιθανών αποτιμήσεων v_i, b_i για τον παίκτη i, θα πρέπει να ισχύει το εξής:

$\mathbf{U_i(v_{-i},v_i\ ;\ v_i) = v_i(\ f(v_{-i},v_i)\ ) - P_i(v_{-i},v_i) \geq U_i(v_{-i},\ b_i\ ;\ v_i) = v_i(\ f(v_{-i},\ b_i)\ ) - P_i(v_{-i},\ b_i)}$	Εξ. 2.2

Δηλαδή, αν v_i είναι η πραγματική συνάρτηση αποτίμησης για τον παίκτη i, τότε όποιο κι αν είναι το προφίλ δηλώσεων συνάρτησης αποτίμησης v_{-i} των άλλων παικτών, η καλύτερη επιλογή του i να είναι να δηλώσει την πραγματική του αποτίμηση v_i αντί οποιασδήποτε άλλης δήλωσης, v_i'.

[3] Μια στρατηγική ενός παίκτη i ονομάζεται κυρίαρχη, όταν είναι η καλύτερη από οποιαδήποτε άλλη στρατηγική που μπορεί να επιλέξει ο παίκτης i, ανεξάρτητα με το πώς παίζουν (τι στρατηγικές επιλέγουν) οι υπόλοιποι παίκτες του παιχνιδιού.

2.4.3. Παραδείγματα Μηχανισμών: Δημοπρασίες Πρώτης και Δεύτερης Τιμής

Αντιπροσωπευτικό παράδειγμα ενός προβλήματος σχεδιασμού μηχανισμών αποτελεί η *Δημοπρασία Δεύτερης Τιμής (Vickrey "second-price" auction)* που μελετήθηκε από τον William Vickrey το 1961 [Vick61].

Συγκεκριμένα, υπάρχει ένα αντικείμενο (αγαθό) για πώληση και πολλοί παίκτες (bidders), καθένας εκ των οποίων προσφέρει μια τιμή (bid, v_i) σε σφραγισμένο φάκελο προκειμένου να αποκτήσει το αγαθό. Ο πλειστηριαστής (auctioneer) ανοίγει όλους τους φακέλους και κατανέμει το αγαθό σε έναν από τους παίκτες, ενώ παράλληλα αποφασίζει και για το τίμημα που καλείται να πληρώσει ο παίκτης που παίρνει το αγαθό. Θα πρέπει λοιπόν να αποφασίσει ο πλειστηριαστής έναν μηχανισμό, του οποίου:

1. Ο κανόνας ανάθεσης θα καθορίζει τον παίκτη που θα αγοράσει το αγαθό.

2. Ο αλγόριθμος κοστολόγησης θα χρεώνει με κάποια τιμή τον παίκτη που αγοράζει το αγαθό, ενώ (προφανώς) δεν χρεώνει καθόλου τους υπόλοιπους παίκτες.

Ας δούμε δυο απλούς μηχανισμούς για δημοπρασίες:

- **Δημοπρασία Μεγαλύτερης Τιμής** (First Price Auction): Το αγαθό δίνεται στον παίκτη i που κάνει την μεγαλύτερη προσφορά, ενώ η τιμή p_i που καλείται να πληρώσει είναι ακριβώς ίση με την προσφορά του. Δηλαδή, αν κάθε παίκτης i έχει μια πραγματική αποτίμηση v_i για το αγαθό, και κάνει μια προσφορά b_i, τότε η συνάρτηση ωφέλειάς του εξαρτάται από το αν κερδίζει ή χάνει το αγαθό, και ορίζεται ως εξής: $U_i(\text{i-wins}) = v_i - b_i$ και $U_i(\text{i-loses}) = 0$.

- **Δημοπρασία Δεύτερης Μεγαλύτερης Τιμής** (Second Price Auction): Το αγαθό δίνεται στον παίκτη i που κάνει την μεγαλύτερη προσφορά, ενώ η τιμή p_i που καλείται να πληρώσει ο παίκτης αυτός είναι ακριβώς ίση με την αμέσως μικρότερη προσφορά (μετά από τη δική του). Δηλαδή, αν κάθε παίκτης i έχει μια

πραγματική αποτίμηση v_i για το αγαθό, και κάνει μια προσφορά b_i, τότε η συνάρτηση ωφέλειάς του εξαρτάται από το αν κερδίζει ή χάνει το αγαθό, και ορίζεται ως εξής: $U_i(\text{i-wins}) = v_i - \max_{k \neq i}\{b_k\}$ και $U_i(\text{i-loses}) = 0$. Η ειδοποιός διαφορά σε αυτή την περίπτωση είναι ότι η τιμή που καλείται να πληρώσει ο νικητής της δημοπρασίας δεν εξαρτάται από τη δική του προσφορά, αλλά από τις προσφορές των αντιπάλων του.

Είναι εύκολο να δει κανείς ότι η Δημοπρασία Μεγαλύτερης Τιμής δεν εξασφαλίζει την φιλαλήθεια των παικτών. Κάθε παίκτης i έχει μια (κρυφή) αποτίμηση v_i της αξίας του αγαθού, ενώ δηλώνει μια αποτίμηση b_i στον πλειστηριαστή. Η ωφέλειά του εξαρτάται από το αν κερδίζει ή όχι το αγαθό: $U_i(\text{i-wins}) = v_i - b_i$ και $U_i(\text{i-loses}) = 0$. Ας δούμε όμως τι συμφέρει το συγκεκριμένο παίκτη να κάνει:

1. Αν δηλώσει $b_i > v_i$, τότε $U_i(\text{i-wins}) = v_i - b_i < 0$, ενώ $U_i(\text{i-loses}) = 0$.

2. Αν δηλώσει $b_i = v_i$, τότε $U_i(\text{i-wins}) = v_i - b_i = 0 = U_i(\text{i-loses})$.

3. Αν δηλώσει $b_i < v_i$, τότε $U_i(\text{i-wins}) = v_i - b_i > 0$, ενώ $U_i(\text{i-loses}) = 0$.

Είναι πλέον προφανές ότι το να δηλώσει προσφορά χαμηλότερη από την πραγματική του πρόθεση είναι η μοναδική περίπτωση που του εξασφαλίζει θετική ωφέλεια (σε περίπτωση που κερδίσει). Άρα, κάθε παίκτης έχει συμφέρον να δηλώσει (όσο το δυνατόν) χαμηλότερη προσφορά από την πραγματική του πρόθεση, και τελικά όλοι οι παίκτες θα δώσουν πρακτικά μηδενική προσφορά για το αγαθό. Ο συγκεκριμένος μηχανισμός λοιπόν ΔΕΝ ΕΙΝΑΙ φιλαλήθης.

Αντίθετα, η Δημοπρασία Δεύτερης Μεγαλύτερης Τιμής εξασφαλίζει την φιλαλήθεια των παικτών. Ας δούμε τι συμφέρει έναν παίκτη $i \in N$ να κάνει:

- Αν δηλώσει $b_i > v_i$, τότε $U_i(\text{i-wins}) = v_i - \max_{k\neq i}\{b_k\}$, που μπορεί να είναι ακόμη και ίσο με $-(b_i - v_i)$ αν υπάρχει και δεύτερος παίκτης με προσφορά $b_k = b_i$, ενώ $U_i(\text{i-loses}) = 0$.

- Αν δηλώσει $b_i = v_i$, τότε $U_i(\text{i-wins}) = v_i - \max_{k\neq i}\{b_k\} \geq 0 = U_i(\text{i-loses})$, αφού νικά μόνο στην περίπτωση που $b_i \geq \max_{k\neq i}\{b_k\}$.

- Αν δηλώσει $b_i < v_i$, τότε υπάρχουν δυο περιπτώσεις:
  (i) Είτε $\max_{k\neq i}\{b_k\} \in [b_i, v_i)$, και έχει ωφέλεια $U_i(\text{i-loses}) = 0$, ενώ θα μπορούσε να αυξήσει την προσφορά του (πχ, $b_i = \max_{k\neq i}\{b_k\} + \varepsilon$, για κάποιο πολύ μικρό $\varepsilon>0$) και να έχει θετική ωφέλεια αφού θα κερδίσει:
  $U_i(\text{i-wins}) = v_i - \max_{k\neq i}\{b_k\} \geq \max_{k\neq i}\{b_k\} + \varepsilon - \max_{k\neq i}\{b_k\} = \varepsilon > 0$.

  (ii) Στην περίπτωση που $\max_{k\neq i}\{b_k\} < b_i$, αυτό σημαίνει ότι ήδη με την προσφορά που έκανε κερδίζει, και η ωφέλειά του είναι ακριβώς $U_i(\text{i-wins}) = v_i - \max_{k\neq i}\{b_k\} > 0$, ανεξάρτητα από την τιμή της δικής του προσφοράς. Ακριβώς την ίδια ωφέλεια όμως θα είχε ακόμη κι αν ήταν ειλικρινής, δηλαδή, είχε κάνει προσφορά $b_i = v_i$.

2.5. Πρόβλημα Ανάθεσης Διεργασιών και Σχεδιασμός Μηχανισμών

Στην εργασία μας, μελετάμε το πρόβλημα ανάθεσης διεργασιών σε μηχανές, το οποίο περιγράφεται ως εξής: Έστω ότι ο μηχανισμός πρέπει να αναθέσει ένα σύνολο διεργασιών n σε ένα σύνολο μηχανών m. Στόχος του είναι να ελαχιστοποιήσει τον χρόνο ολοκλήρωσης της τελευταίας διεργασίας (το makespan) που θα ανατεθεί. Ο μηχανισμός, προκειμένου να δώσει λύση στο πρόβλημα, πρέπει να γνωρίζει τους χρόνους εκτέλεσης των διεργασιών στις εκάστοτε μηχανές. Όμως, τους χρόνους αυτούς δεν τους γνωρίζει. Με σκοπό να διαχειριστεί την έλλειψη πληροφορίας, αντιμετωπίζει το πρόβλημα ως παιχνίδι και τις μηχανές ως παίκτες. Σκοπός είναι να σχεδιάσει ένα

παιχνίδι, οι κανόνες του οποίου θα αναγκάσουν τους παίκτες (μηχανές) να δηλώσουν τους πραγματικούς χρόνους που χρειάζονται για να εκτελέσουν τις διεργασίες.

2.5.1. Ορισμός Προβλήματος Ανάθεσης Διεργασιών

Έστω ότι έχουμε n διεργασίες που πρέπει να ανατεθούν σε m μηχανές (παίκτες). Ο *τύπος*[4] κάθε παίκτη i καθορίζεται από το διάνυσμα $t_i = (t_{ij})_{j\in[n]} \in R_{>0}$ των πραγματικών χρόνων εκτέλεσης t_{ij} για κάθε διεργασία j και αποτελεί μια κρυφή πληροφορία του παίκτη, δηλαδή *είναι γνωστή μόνο στον παίκτη i. Οτιδήποτε άλλο στο πρόβλημα είναι δημοσίως γνωστό.* Στόχος του συστήματος είναι η επιλογή ανάθεσης των διεργασιών στις μηχανές ώστε να ελαχιστοποιείται ο ***μέγιστος χρόνος περάτωσης διεργασίας (makespan),*** δηλαδή, το μέγιστο (ως προς όλες τις μηχανές) άθροισμα χρόνων εκτέλεσης των διεργασιών που ανατίθενται σε μια μηχανή. Αντίθετα, κάθε παίκτης (μηχανή) αποτιμά μια ανάθεση ανάλογα με το συνολικό χρόνο εκτέλεσης που απαιτεί από αυτήν. Δηλαδή, η συνάρτηση αποτίμησης του παίκτη i είναι η αρνητική τιμή του συνολικού χρόνου που δαπανά για να εκτελέσει τις διεργασίες που του έχουν ανατεθεί. Πιο συγκεκριμένα[NiRo99]:

- $A = \{ \alpha = (\alpha_{ij}) \in \{0,1\}^{m\times n} : \forall j\in[n], \Sigma_{i\in[m]} \alpha_{ij} = 1\}$ είναι το σύνολο των δυνατών αναθέσεων (αναπαριστώνται μέσω των χαρακτηριστικών διανυσμάτων α) των διεργασιών στις μηχανές. Για ένα συγκεκριμένο διάνυσμα $\alpha\in A$ ανάθεσης των διεργασιών στις μηχανές, συμβολίζουμε με α_i το επιμέρους διάνυσμα που υποδεικνύει την ανάθεση διεργασιών στη μηχανή i.

- Κάθε παίκτης $i\in N$ έχει έναν τύπο $t_i = (t_{ij})_{j\in[n]} \in R_{>0}$ που καθορίζει τους χρόνους εκτέλεσης των διεργασιών από αυτόν. Για μια οποιαδήποτε ανάθεση $\boldsymbol{\alpha}\in A$, ο παίκτης i επιβαρύνεται με ένα συνολικό **κόστος εκτέλεσης** $\Sigma_{j\in[m]} \alpha_{ij}\cdot t_{ij}$ των

[4] Ο τύπος t_n ενός παίκτη (μηχανής) n είναι η κρυφή πληροφορία του, που υποδηλώνει ΚΑΙ την (επίσης κρυφή) συνάρτηση αποτίμησης v_n του παίκτη αυτού.

διεργασιών που του ανατίθενται, σύμφωνα με την ανάθεση **α**. Η *συνάρτηση αποτίμησης* του παίκτη i είναι λοιπόν: $\mathbf{v_i(\alpha,t_i) = -\Sigma_{j\in[m]}\ \alpha_{ij}\cdot t_{ij}}$

- Η *αντικειμενική συνάρτηση* (objective function), που επιθυμούμε να ελαχιστοποιήσουμε, είναι η εξής: $\mathbf{g(\alpha,t) = max_{i\in[m]}\ \{\ \Sigma_{j\in[m]}\ \alpha_{ij}\cdot t_{ij}\ \}}$

- Ο μηχανισμός αποτελείται από δυο μέρη:
    1. Τη **συνάρτηση ανάθεσης f** : $T \rightarrow A$ που δέχεται ως είσοδο ένα διάνυσμα δηλώσεων τύπων $b\in T$ από τους παίκτες, και επιλέγει μια συγκεκριμένη ανάθεση $f(b) \in A$ των διεργασιών στις μηχανές. Με $f_i(b)$ συμβολίζουμε το διάνυσμα που υποδεικνύει ποιες διεργασίες ανατίθενται στη μηχανή i. Με $f_{ij}(b)$ συμβολίζουμε την ενδεικτική μεταβλητή που διευκρινίζει αν στην ανάθεση f(b) η διεργασία j ανατίθεται στην μηχανή i.

    2. Τις **συναρτήσεις ανταμοιβής** (payment functions): Κάθε μηχανή i, ως ανταμοιβή για την εκτέλεση των διεργασιών $f_i(b)$ που αναλαμβάνει όταν οι μηχανές δηλώνουν τους τύπους σύμφωνα με το προφίλ τύπων b, λαμβάνει κάποια πληρωμή $P_i(b)$, που είναι η συνάρτηση κοστολόγησης του μηχανισμού.

Στον παραπάνω ορισμό η αντικειμενική συνάρτηση g μας δίνει το *makespan* για κάθε ανάθεση $\alpha\in A$, δεδομένων των πραγματικών τύπων των μηχανών. Στόχος μας είναι να βρούμε εκείνη την ανάθεση α* που θα ελαχιστοποιεί το makespan, δεδομένων των πραγματικών τύπων των μηχανών. Δηλαδή, η συνάρτηση κοινωνικής επιλογής που υιοθετούμε, είναι η εξής: $\mathbf{scf(t) = \alpha^* \in arg\ max_{\alpha\in A}\ g(\alpha,t)}$

Οι μηχανές είναι οι εγωιστικοί παίκτες. Άρα, αρκεί μια μηχανή i να δώσει το διάνυσμα b_i των δικών της (δηλώσεων) χρόνων εκτέλεσης, προκειμένου να καθορίσει τη δική της (γραμμική ως προς τα φορτία που της ανατίθενται) συνάρτηση αποτίμησης.

Η συνάρτηση ωφέλειας του παίκτη i, για τη δήλωση τύπων b από όλους τους παίκτες, δεδομένου ότι οι πραγματικοί χρόνοι εκτέλεσης της μηχανής i δίνονται από τον πραγματικό τύπο t_i, είναι η εξής:

$$\mathbf{U_i\,(b_{-i},b_i\,;\,t_i) = -(\,t_i \cdot f_i(b_{-i},b_i)\,) + P_i(b_{-i},b_i) = P_i(b_{-i},b_i) - \Sigma_{j\in[n]}\,[t_{ij}\cdot f_{ij}(b_{-i},b_i)]}$$

Η συνάρτηση ωφέλειας εκφράζει το συνολικό όφελος που αποκομίζει ο παίκτης από μια συγκεκριμένη ανάθεση των διεργασιών (αποτέλεσμα/ έξοδος : α = f(b)). Οι παίκτες είναι ιδιοτελείς, και συνεπώς επιθυμούν να εργαστούν όσον το δυνατό λιγότερο και να πληρωθούν όσο το δυνατό περισσότερο.

2.5.2. Παράδειγμα Προβλήματος Ανάθεσης Διεργασιών

Έστω ότι έχουμε 2 μηχανές (παίκτες) $\{M_1, M_2\}$ στις οποίες θέλουμε να αναθέσουμε 3 διεργασίες $\{j_1, j_2, j_3\}$. Στον παρακάτω πίνακα δίνονται οι χρόνοι εκτέλεσης των διεργασιών.

	j_1	j_2	j_3
M_1	20	40	50
M_2	70	30	60

Τα εφικτά αποτελέσματα του προβλήματος είναι όλες οι πιθανές αναθέσεις των διεργασιών στις 2 μηχανές. Ας υποθέσουμε ότι ο μηχανισμός γνωρίζει τους πραγματικούς χρόνους εκτέλεσης των διεργασιών (δηλαδή, τους πραγματικούς τύπους των μηχανών) με βάση τους οποίους υπολογίζει, για κάθε δυνατή ανάθεση, το makespan μέσω της συνάρτησης g. Από τα αποτελέσματα της g επιλέγει εκείνη την ανάθεση για την οποία η g δίνει την ελάχιστη τιμή. Αυτή θα είναι και η τελική ανάθεση των

διεργασιών στις μηχανές που θα επέλεγε η συνάρτηση κοινωνικής επιλογής scf. Για το παράδειγμα μας η καλύτερη ανάθεση είναι η M_1: $\{j_1, j_2\}$ και M_2: $\{j_3\}$ με makespan 60.

Στο παράδειγμα υποθέσαμε ότι ο μηχανισμός γνωρίζει τους χρόνους εκτέλεσης των διεργασιών. Κάτι τέτοιο όμως δεν είναι δεδομένο εκ των προτέρων. Αν ήταν τότε δεν θα χρειαζόταν κάποιος μηχανισμός για να βρούμε την βέλτιστη ανάθεση. Απλά θα χρησιμοποιούσαμε κάποιο βέλτιστο αλγόριθμο για το χρονοπρογραμματισμό διεργασιών. Θεωρούμε ότι κάθε παίκτης και μόνο αυτός γνωρίζει τους χρόνους στους οποίους μπορεί να εκτελέσει τις διεργασίες. Ο βασικός λόγος σχεδίασης του μηχανισμού είναι ότι ο σχεδιαστής δεν γνωρίζει τους χρόνους αυτούς. Οι παίκτες θεωρούνται ιδιοτελείς και είναι πιθανό να δηλώσουν ψευδείς χρόνους για να αναλάβουν λιγότερες διεργασίες. Ο σχεδιαστής πρέπει να επιλέξει εκείνο το μηχανισμό που θα «αναγκάζει» τους παίχτες να του αποκαλύπτουν τους πραγματικούς χρόνους εκτέλεσης, και παράλληλα θα καταφέρνει να προσδιορίσει μια ανάθεση της οποίας το makespan απέχει όσο το δυνατόν λιγότερο από το makespan της βέλτιστης ανάθεσης που θα επέστρεφε η scf (ως προς τους πραγματικούς χρόνους εκτέλεσης). Μας ενδιαφέρει επίσης ο προτεινόμενος μηχανισμός να είναι και αποδοτικός ως προς την πολυπλοκότητά του.

2.6. Στόχοι Εργασίας

Συνοψίζοντας, ένας μηχανισμός είναι ο προσδιορισμός των κανόνων ενός παιχνιδιού με τέτοιο τρόπο ώστε: (i) οι συμμετέχοντες στο παιχνίδι να είναι φιλαλήθεις ως προς τις προθέσεις τους, και (ii) η λύση που θα προκύψει ως αποτέλεσμα του παιχνιδιού να είναι όσο το δυνατόν πλησιέστερα στην βέλτιστη λύση που θα επέλεγε ο ίδιος ο σχεδιαστής του μηχανισμού αν είχε πλήρη έλεγχο, προς όφελος ολόκληρου του συστήματος (όπως αυτό περιγράφεται μέσα από μια συνάρτηση κοινωνικής ωφέλειας). Στην εργασία ασχολούμαστε με αλγοριθμικά ζητήματα σχεδιασμού μηχανισμών.

Συγκεκριμένα, μελετάμε το πρόβλημα της εύρεσης (αποδοτικών) μηχανισμών, αλλά και κάτω φραγμάτων, που σχετίζονται με το πρόβλημα της ανάθεσης διεργασιών (task scheduling) σε παράλληλες μηχανές. Στο συγκεκριμένο σενάριο, κάθε μηχανή θεωρείται ότι έχει τους δικούς της χρόνους εκτέλεσης των διεργασιών, και αυτή είναι προσωπική (κρυφή) πληροφορία κάθε μηχανής. Ο σχεδιαστής επιθυμεί να αναθέσει τις διεργασίες στις μηχανές με τέτοιο τρόπο ώστε να ελαχιστοποιείται ο χρόνος περάτωσης και της τελευταίας διεργασίας (makespan). Όμως δεν είναι σε θέση να το κάνει αυτό, γιατί δεν γνωρίζει τους πραγματικούς χρόνους εκτέλεσης των διεργασιών στις μηχανές. Ζητά λοιπόν από τις μηχανές να αποκαλύψουν τα διανύσματα των χρόνων εκτέλεσης των διεργασιών, και στη συνέχεια, με βάση αυτή την πληροφορία : (i) επιλέγει μια ανάθεση των διεργασιών στις μηχανές, και (ii) ανταμείβει κάθε μηχανή για την συνεισφορά της στην εκτέλεση των διεργασιών. Όμως, κάθε (εγωιστικά συμπεριφερόμενη) μηχανή στοχεύει στο να ελαχιστοποιήσει τη δική της προσπάθεια (χρόνο εκτέλεσης των διεργασιών που της ανατίθενται) και να μεγιστοποιήσει την ανταμοιβή που θα λάβει. Για το λόγο αυτό ενδέχεται να δώσει εσφαλμένα δεδομένα στο σχεδιαστή. Στόχος λοιπόν του σχεδιαστή του μηχανισμού είναι να διαλέξει τους κατάλληλους αλγόριθμους ανάθεσης διεργασιών, και καθορισμού των ανταμοιβών, ώστε: (i) οι μηχανές να μην έχουν όφελος δίνοντας ψευδή στοιχεία για τους χρόνους εκτέλεσης, και (ii) η ανάθεση που θα προκύψει να έχει makespan όσο το δυνατόν πλησιέστερο στο makespan της βέλτιστης λύσης που θα υπολόγιζε ο σχεδιαστής, λαμβάνοντας υπόψη τους πραγματικούς χρόνους εκτέλεσης. Προκειμένου ο μηχανισμός να είναι αποδοτικός, αυτοί οι δυο αλγόριθμοι θα πρέπει να είναι πολυωνυμικού χρόνου ως προς το μέγεθος του προβλήματος.

Οι μηχανισμοί που αναλύουμε εξασφαλίζουν ότι οι παίκτες δηλώνουν με φιλαλήθεια τους πραγματικούς χρόνους επεξεργασίας των διεργασιών. Στην εργασία μας, γίνεται προσπάθεια να αναλύσουμε τις ιδιότητες που έχουν οι φιλαλήθεις μηχανισμοί, βασιζόμενοι κυρίως στο θεώρημα της μονοτονικότητας των Archer και Tardos.

Μελετάμε το συγκεκριμένο πρόβλημα (επιλογής ενός αποδοτικού μηχανισμού για ανάθεση διεργασιών σε μηχανές) σε τρεις βασικές κατηγορίες: σε παράλληλες

συσχετιζόμενες μηχανές (uniformly related machines), σε μηχανές με περιορισμό ως προς την πρόσβαση (restricted assignment machines), και σε μη συσχετιζόμενες μηχανές (unrelated machines).

Στο επόμενο κεφάλαιο περιγράφεται η ιδέα της φιλαλήθειας (truthfulness) και δίνονται κάποιοι τρόποι, γενικές αρχές και τεχνικές που οδηγούν στη δημιουργία φιλαληθών (truthful) μηχανισμών.

ΚΕΦΑΛΑΙΟ 3. Η ΙΔΙΟΤΗΤΑ ΤΗΣ ΦΙΛΑΛΗΘΕΙΑΣ (TRUTHFULNESS)

3.1. Ορισμός Φιλαλήθειας

Ο σχεδιαστής του μηχανισμού πρέπει να είναι σίγουρος ότι οι παίκτες που συμμετέχουν στο παιχνίδι είναι φιλαλήθεις, πράγμα που εξασφαλίζει σχεδιάζοντας κατάλληλα το μηχανισμό (μέσω των κανόνων ανάθεσης). Θεωρούμε ότι οι παίκτες έχουν άπειρη υπολογιστική δύναμη, συνεπώς μπορούν να μελετήσουν πλήρως τον μηχανισμό καθώς και τα αποτελέσματα που θα δώσει ο αλγόριθμος ανάθεσης για κάθε πιθανή στρατηγική των παικτών. Επιπλέον οι παίκτες θεωρούμε ότι είναι λελογισμένοι και πάντα επιλέγουν την καλύτερη στρατηγική για το παιχνίδι.

Προκειμένου να είναι οι παίκτες φιλαλήθεις, πρέπει το παιχνίδι να σχεδιαστεί έτσι ώστε οι παίκτες μελετώντας το να καταλήγουν στο συμπέρασμα ότι το όφελός τους μεγιστοποιείται μόνο αν είναι φιλαλήθειας και σε καμία άλλη περίπτωση. Δηλαδή να είναι η στρατηγική της φιλαλήθειας η καλύτερη στρατηγική γι' αυτούς (κυρίαρχη στρατηγική-dominant strategy). Αν επιτευχθεί αυτό τότε το παιχνίδι καταλήγει σε ένα επιθυμητό σημείο ισορροπίας (κατάσταση ισορροπίας Nash) όπου όλοι οι παίκτες έχουν επιλέξει την στρατηγική της φιλαλήθειας και έχουν πλεονέκτημα να παραμείνουν σε

αυτή. Ένας τέτοιος μηχανισμός καλείται φιλαλήθης και απαιτείται αφενός για να επιτυγχάνεται δικαιοσύνη στο σύστημα και αφετέρου για να γνωρίζουμε ποιο πρόβλημα λύνουμε, δηλαδή οι συναρτήσεις αποτίμησης πρέπει να έχουν μια πληροφορία, γιατί αν είναι ψευδείς ο μηχανισμός τελικά θα δώσει ασήμαντα αποτελέσματα.

Στις επόμενες παραγράφους, περιγράφονται κάποιοι τρόποι, γενικές αρχές και τεχνικές που οδηγούν στη δημιουργία φιλαληθών μηχανισμών.

3.2. Η Αρχή της Αποκάλυψης (Revelation Principle)

ΟΡΙΣΜΟΣ 3.1 [Μηχανισμός Άμεσης Αποκάλυψης]: Μια συνάρτηση κοινωνικής επιλογής (social choice function): $f : V_1 \times ... \times V_n \to A$ αποτελεί ένα μηχανισμό άμεσης αποκάλυψης (direct revelation mechanism). Το διάνυσμα κοστολόγησης $P_1,\ldots,P_n$ είναι το ποσό που πρέπει να πληρώσει ο παίκτης i.

Στους μηχανισμούς άμεσης αποκάλυψης οι παίκτες δεν επικοινωνούν μεταξύ τους αλλά με έναν μεσάζοντα (τον σχεδιαστή του μηχανισμού), στον οποίο δηλώνουν εμπιστευτικά όλη την κρυφή τους πληροφορία, που αφορά την προτίμηση που έχουν οι ίδιοι όσον αφορά όλα τα δυνατά αποτελέσματα του μηχανισμού (καταστάσεις συστήματος). Ο μεσάζοντας, αφού λάβει όλη την απαραίτητη πληροφορία, καθορίζει όλες τις εφικτές καταστάσεις του συστήματος (π.χ. κατανομές αγαθών, αναθέσεις διεργασιών κ.λπ.) συναρτήσει των δηλώσεων των παικτών. Ο σχεδιαστής του μηχανισμού, μέσω του μηχανισμού άμεσης αποκάλυψης, καθορίζει τους κανόνες που εξασφαλίζουν ότι οι παίκτες απαντούν ειλικρινά σε ότι αφορά τις δηλώσεις τους.

ΘΕΩΡΗΜΑ 3.2 [Αρχή Αποκάλυψης]: Αν υπάρχει ένας μηχανισμός M που υλοποιεί την f με κυρίαρχες στρατηγικές, τότε υπάρχει κι ένας φιλαλήθης μηχανισμός άμεσης αποκάλυψης M΄ που υλοποιεί την f.

Απόδειξη: Έστω $a_i(t_i)$ συμβολίζει μια κυρίαρχη στρατηγική ενός παίκτη i όταν ο τύπος του είναι t_i. (μηχανισμός M). Προσομοιώνοντας τον μηχανισμό M, ορίζουμε έναν νέο μηχανισμό άμεσης αποκάλυψης M΄ για τον οποίο ισχύει $f(t_i) = s(a_i(t_i))$ και $P_i'(t_i) = P_i(a_i(t_i))$. Αφού $a_i(t_i)$ είναι κυρίαρχη στρατηγική για τον παίκτη i, τότε για κάθε t_i, $a_{-i}(t_{-i})$, $a_i(t_i')$ έχουμε

$$v_i(t_i, s(a_i(t_i), a_{-i}(t_{-i})) - P_i(a_i(t_i), a_{-i}(t_{-i})) \geq v_i(t_i, s(a_i(t_i'), a_{-i}(t_{-i})) - P_i(a_i(t_i'), a_{-i}(t_{-i}))$$

που δίνει τον ορισμό της φιλαλήθειας για τον μηχανισμό M΄.

Σε ένα μηχανισμό άμεσης αποκάλυψης οι παίκτες έχουν μόνο μία στρατηγική, ¨αποκαλύπτουν τις τιμές τους στον μηχανισμό¨. Επομένως, ο μηχανισμός θα είναι φιλαλήθης όταν η στρατηγική της φιλαλήθειας είναι κυρίαρχη, και δίνει στους παίκτες τα καλύτερα αποτελέσματα.

3.3. Θεώρημα Gibbard-Satterthwaite

Έστω ότι θέλουμε να δημιουργήσουμε φιλαλήθεις μηχανισμούς χωρίς να χρησιμοποιήσουμε το σχήμα κοστολόγησης (δηλαδή τις αποζημιώσεις), που μπορεί να επιτύχει κάτι τέτοιο (βλ. παράγραφο 3.4). Σε αυτήν την περίπτωση, η μόνη χρήσιμη πληροφορία που προκύπτει από τη συνάρτηση ωφέλειας U_i, είναι η αποτίμηση που έχει ο παίκτης για κάθε πιθανό αποτέλεσμα ανάθεσης. Έτσι, κατατάσσοντας αυτές τις αποτιμήσεις κατά φθίνουσα σειρά, παίρνουμε μια διάταξη των αποτιμήσεων του παίκτη i (relative order of agent i's preferences).

Οι διατεταγμένες αποτιμήσεις του παίκτη δε συσχετίζονται με κάποιο τρόπο μεταξύ τους: έστω ότι από τον μηχανισμό επιλέγεται το αποτέλεσμα Y ως προτιμότερο για τον παίκτη i, δεδομένου του τύπου του t_i. Μια μικρή τροποποίηση στον τύπο του παίκτη, η οποία αυξάνει την επιθυμία του για το αποτέλεσμα X, μπορεί είτε να μην προκαλέσει

αλλαγή στο αποτέλεσμα του μηχανισμού, είτε να επιλεχθεί το αποτέλεσμα X ως το πλέον προτιμότερο για τον παίκτη i.

Επομένως (και σύμφωνα με τον συμβολισμό της παραγράφου 2.4.1), η συνάρτηση ωφέλειας μπορεί να αναπαρασταθεί με μια διάταξη των αποτιμήσεων του παίκτη $\succ_i$. Ο ισχυρισμός $x \succ_i y$, όπου x, y ϵ A, σημαίνει ότι ο παίκτης i επιθυμεί το αποτέλεσμα x τουλάχιστον τόσο όσο επιθυμεί και το αποτέλεσμα y, δηλαδή $u_i(x) \geq_i u_i(y)$. Η διάταξη των προτιμήσεων του παίκτη i είναι *¨αυστηρή¨ (strict)* αν δεν υπάρχουν δύο αποτελέσματα που να είναι το ίδιο επιθυμητά από αυτόν. Δηλαδή, ισχύει $x \succ_i y$ ή $y \succ_i x$.

Συμβολίζουμε με P το προφίλ ενός παίκτη, που είναι το σύνολο όλων των ¨αυστηρών¨ διατάξεων των προτιμήσεών του επί των ενδεχόμενων καταστάσεων A που μπορεί να περιέλθει το σύστημα. Ουσιαστικά, πρόκειται για ένα *διάνυσμα από διατάξεις προτιμήσεων (προφίλ προτιμήσεων $\succ$)* που υποθέτουμε ότι ισοδυναμεί με τον τύπο του παίκτη $T_i = P$, και άρα $T = P^N$.

ΟΡΙΣΜΟΣ 3.3 [Δικτατορία]: Μια συνάρτηση κοινωνικής επιλογής είναι δικτατορική (dictatorial) αν υπάρχει ένας παίκτης i ϵ N τέτοιος ώστε για κάθε προφίλ προτιμήσεων $\succ$, να προκύπτει $f(\succ) = a$, όπου "a" το αποτέλεσμα που κατατάσσεται πρώτο στη σχέση $\succ_i$.

Ένας τέτοιος *δικτατορικός κανόνας* αγνοεί τις προτιμήσεις όλων των παικτών εκτός του i, και επομένως επιλέγει το αποτέλεσμα που προτιμάει περισσότερο ο παίκτης i. Ο παίκτης i ονομάζεται δικτάτορας. Η *δικτατορία (dictatorship)* είναι ένα είδος φιλαλήθη μηχανισμού, αφού για όλους τους παίκτες $j \neq i$, όλες οι στρατηγικές είναι το ίδιο κακές, εκτός του παίκτη i που επωφελείται μόνο αν πει την αλήθεια.

ΘΕΩΡΗΜΑ 3.4 [Gibbard-Satterthwaite]: Ένας φιλαλήθης μηχανισμός με τουλάχιστον τρία πιθανά αποτελέσματα είναι δικτατορικός.

3.4. Vickrey-Clarke-Groves (VCG) Μηχανισμός

Σε αντίθεση με το θεώρημα Gibbard-Satterthwaite, στο VCG μηχανισμό χρησιμοποιούμε τις ιδιότητες του σχήματος κοστολόγησης. Ο VCG μηχανισμός αποτελεί μια γενική τεχνική για τη δημιουργία φιλαληθών μηχανισμών.

ΟΡΙΣΜΟΣ 3.5: Ένας μηχανισμός άμεσης αποκάλυψης M = (f(t), P(t)) ανήκει στην οικογένεια των VCG μηχανισμών εάν ισχύουν οι παρακάτω συνθήκες:

1. $f(v_1,...,v_n) \in \arg\max_{a\in A}(\sum_i v_i(a))$
2. για κάποιες αυθαίρετες συναρτήσεις $h_1,\ldots,h_n$, όπου $h_i : V_{-i} \to R$ (το h_i δεν εξαρτάται από το v_i), έχουμε για όλα τα $v_1 \in V_1,...,v_n \in V_n$: $P_i(v_1,...,v_n) = h_i(v_{-i}) - \sum_{j\neq i} v_j(f(v_1,...,v_n))$

Στον παραπάνω ορισμό η συνθήκη 1 υποδηλώνει ότι το αποτέλεσμα του μηχανισμού πάντα θα μεγιστοποιεί την αντικειμενική συνάρτηση του προβλήματος που είναι το άθροισμα των ωφελειών όλων των παικτών.

Η συνθήκη 2 δηλώνει ότι η αποζημίωση κάθε παίκτη i δεν εξαρτάται από την συνάρτηση ωφέλειάς του, αλλά από τις ωφέλειες των άλλων παικτών, καθώς και από μία αυθαίρετη συνάρτηση $h_i(t_{-i})$, η οποία έχει ως πεδίο ορισμού το σύνολο των τύπων των άλλων παικτών, δηλαδή οι τιμές τις δεν εξαρτώνται από τον τύπο του i. Όταν ο όρος $\sum_{j\neq i} v_j(f(v_1,...,v_n))$ προστίθεται στην ωφέλεια του i : $v_i(f(v_1,...,v_n))$, το συνολικό άθροισμα γίνεται ίσο με το άθροισμα των ωφελειών όλων των παικτών της $f(v_1,...,v_n)$.

Επομένως, ο παίκτης αποκτάει κίνητρο να μεγιστοποιήσει τη συνολική κοινωνική ωφέλεια, πράγμα που συμβαίνει όταν είναι φιλαλήθης.

Αυτή ακριβώς η μορφή των συναρτήσεων κοστολόγησης των VCG μηχανισμών είναι που δίνει το ζητούμενο αποτέλεσμα, δηλαδή φιλαλήθεις μηχανισμούς. Αν ο σχεδιαστής εξασφαλίσει αυτό το σχήμα κοστολόγησης μπορεί να είναι σίγουρος ότι οι πληροφορίες που θα πάρει από τους παίκτες είναι αληθινές.

ΘΕΩΡΗΜΑ 3.6 [Φιλαλήθεια VCG Μηχανισμού]: Ένας VCG μηχανισμός είναι φιλαλήθης

Απόδειξη: Έστω ότι η πραγματική αποτίμηση του παίκτη i είναι v_i, η αξία των άλλων παικτών είναι v_{-i} και έστω v_i' μια ψευδής δήλωση του παίκτη i. Θα πρέπει να δείξουμε ότι η ωφέλεια u_i του παίκτη i όταν ο παίκτης δηλώνει v_i δεν είναι μικρότερη από την ωφέλεια u_i' που προκύπτει όταν ο παίκτης δηλώνει v_i'. Δείχνουμε με $a = f(v_i, v_{-i})$ και $a' = f(v_i', v_{-i})$ τα αντίστοιχα αποτελέσματα. Η ωφέλεια του παίκτη i, όταν δηλώνει v_i είναι $u_i = v_i(a) + \Sigma_{j\neq i} v_j(a) - h_i(v_{-i})$ και όταν δηλώνει v_i' είναι $u_i' = v_i(a') + \Sigma_{j\neq i} v_j(a') - h_i(v_{-i})$. Αλλά, αφού το αποτέλεσμα $a = f(v_i, v_{-i})$ μεγιστοποιεί την αντικειμενική συνάρτηση του προβλήματος (συνθήκη 1), ισχύει ότι $v_i(a) + \Sigma_{j\neq i} v_j(a) - h_i(v_{-i}) \geq v_i(a') + \Sigma_{j\neq i} v_j(a') - h_i(v_{-i})$. Αφαιρώντας τον ίδιο όρο $h_i(v_{-i})$ και από τα δύο μέρη της ανισότητας, προκύπτει ότι $v_i(a) + \Sigma_{j\neq i} v_j(a) \geq v_i(a') + \Sigma_{j\neq i} v_j(a')$, που εξασφαλίζει την φιλαλήθεια των παικτών.

3.5. Θεώρημα του Roberts

ΟΡΙΣΜΟΣ 3.7 [Weak Monotonicity (WMON)]: Μια συνάρτηση κοινωνικής επιλογής f ικανοποιεί τη συνθήκη Weak Monotonicity εάν για όλα τα i και v_{-i} ισχύει $f(v_i, v_{-i}) = a \neq b = f(v_i', v_{-i})$ τότε θα ισχύει και $v_i(a) - v_i'(a) \geq v_i(b) - v_i'(b)$.

Σύμφωνα με τη συνθήκη WMON, αν η συνάρτηση κοινωνικής επιλογής μεταβάλλεται από *a* σε b, επειδή ένας παίκτης i μεταβάλλει την δήλωσή του από v_i σε v_i', τότε αυτό σημαίνει ότι ο παίκτης έχει συγκριτικά μεγαλύτερη αξία λόγω της νέας του προτίμησης b, σε σχέση με την αξία που είχε από την παλιά του επιλογή *a*. Δηλαδή, το αποτέλεσμα του μηχανισμού μεταβάλλεται μόνο αν αυτό σημαίνει ότι θα επωφεληθεί ο παίκτης.

ΘΕΩΡΗΜΑ 3.8 [WMON]: Αν ένας μηχανισμός (f, P) είναι φιλαλήθης, τότε η συνάρτηση f ικανοποιεί τη συνθήκη WMON.

Απόδειξη: Αφού ο μηχανισμός είναι φιλαλήθης, τα κόστη $P = (P_1,..,P_n)$ θα εξαρτώνται από το αποτέλεσμα του μηχανισμού και όχι από τις αποτιμήσεις v_i των παικτών. Επομένως, όταν το αποτέλεσμα είναι a ο παίκτης πληρώνει πάντα p_a. Έστω ότι ισχύει $f(v_i, v_{-i}) = a \neq b = f(v_i', v_{-i})$. Αν ο παίκτης με αξία v_i δεν επιθυμεί να δηλώσει v_i' σημαίνει ότι $v_i(a) - p_a \geq v_i(b) - p_b$. Παρομοίως, αν ο παίκτης με αξία v_i' δεν επιθυμεί να δηλώσει v_i σημαίνει ότι $v_i'(a) - p_a \leq v_i'(b) - p_b$. Αφαιρώντας τη δεύτερη ανισότητα από την πρώτη, παίρνουμε $v_i(a) - v_i'(a) \geq v_i(b) - v_i'(b)$.

Το πρόβλημα με τη συνθήκη WMON είναι ότι ισχύει για κάθε παίκτη χωριστά και για κάθε v_{-i} χωριστά, δηλαδή είναι μια τοπική συνθήκη (local condition). Η ύπαρξη μιας ολικής συνθήκης (global condition) εξαρτάται από το πεδίο ορισμού των αποτιμήσεων V_i των παικτών. Ολική συνθήκη υπάρχει για τις δύο ακραίες περιπτώσεις όπου α) το πεδίο ορισμού V_i είναι ¨μη περιορισμένο¨ (unrestricted) και β) το πεδίο ορισμού V_i είναι ¨αυστηρά περιορισμένο¨ (severely restricted/ single dimensional), δηλαδή υπάρχει μια πραγματική παράμετρος που καθορίζει ολόκληρο το διάνυσμα v_i. Στο ενδιάμεσο διάστημα όπου το πεδίο V_i είναι ¨περίπου περιορισμένο¨, η εύρεση μιας ολικής συνθήκης, η οποία ικανοποιείται όταν ο μηχανισμός είναι φιλαλήθης, αποτελεί ανοικτό πρόβλημα.

Το πεδίο ορισμού V_i είναι ¨μη περιορισμένο¨: αποδεικνύεται ότι σε αυτήν την περίπτωση οι μόνοι φιλαλήθεις μηχανισμοί είναι απλές παραλλαγές των VCG

μηχανισμών. Αυτές οι παραλλαγές προϋποθέτουν βάρη για τους παίκτες, βάρη για τα πιθανά αποτελέσματα, και επιτρέπουν τον περιορισμό του πεδίου τιμών της συνάρτησης $f(v_i,\ldots,v_n)$. Η συνάρτηση κοινωνικής επιλογής που προκύπτει είναι ¨affine maximizer¨.

ΟΡΙΣΜΟΣ 3.9 [Affine Maximizer]: Μια συνάρτηση κοινωνικής επιλογής f είναι affine maximizer αν για κάποιο υποσύνολο A΄ του συνόλου A (A΄⊂ A), για κάποια βάρη $w_1,\ldots w_n \in R^+$ των παικτών και για κάποια βάρη $c_a \in R$ των αποτελεσμάτων a ϵ A΄, ισχύει ότι $f(v_1,\ldots,v_n) \in argmax_{a\epsilon A'}(c_a + \Sigma_i w_i v_i(a))$.

Παρακάτω δείχνουμε πως οι VCG μηχανισμοί μπορούν να γενικευθούν σε affine maximizers:

ΠΡΟΤΑΣΗ 3.10: Έστω ότι η συνάρτηση f είναι affine maximizer. Τα κόστη, για κάθε παίκτη i, ορίζονται ως εξής $P_i(v_1,\ldots,v_n) = h_i(v_{-i}) - 1/w_i[\Sigma_{j\neq i} w_j v_j(a) - c_a]$, όπου h_i μια αυθαίρετη συνάρτηση που δεν εξαρτάται από το v_i. Ο μηχανισμός (f, P) είναι φιλαλήθης.

Απόδειξη: Χωρίς βλάβη της γενικότητας μπορούμε να υποθέσουμε ότι $h_i = 0$. Εάν προκύψει το αποτέλεσμα a, η ωφέλεια του παίκτη i θα ισούται με $v_i(a) + 1/w_i[\Sigma_{j\neq i} w_j v_j(a) + c_a]$. Αν πολλαπλασιάσουμε με $w_i > 0$, η παραπάνω συνάρτηση θα μεγιστοποιείται όταν μεγιστοποιείται και η ποσότητα $c_a + \Sigma_j w_j v_j(a)$, πράγμα που συμβαίνει όταν ο παίκτης i δηλώνει την πραγματική του αξία v_i.

ΘΕΩΡΗΜΑ 3.11 [Roberts]: Αν $|A| \geq 3$, η f είναι επί του συνόλου A, $V_i = R^A$ για κάθε i, και ο μηχανισμός (f, P) είναι φιλαλήθης τότε η f είναι affine maximizer.

ΚΕΦΑΛΑΙΟ 4. ΑΝΑΘΕΣΗ ΔΙΕΡΓΑΣΙΩΝ ΣΕ ΠΑΡΑΛΛΗΛΑ ΣΥΣΧΕΤΙΖΟΜΕΝΕΣ ΜΗΧΑΝΕΣ

4.1. Πρόβλημα Ανάθεσης Διεργασιών σε Συσχετιζόμενες Μηχανές

Το πρόβλημα Q| |C_{max} ορίζεται ως εξής:

Επιθυμούμε την ανάθεση ενός συνόλου διεργασιών, N = {1,2,...,n}, κάθε μια από τις οποίες έχει κάποιο **φόρτο εξυπηρέτησης** $p_j > 0$, με αδιαίρετο τρόπο, σε ένα σύνολο M={1,2,…,m} από **ομοιόμορφα συσχετιζόμενες μηχανές** (uniformly related machines), καθεμιά από τις οποίες έχει τη δική της **ταχύτητα εξυπηρέτησης, s_i**. Έτσι, αν η διεργασία j ανατεθεί στη μηχανή i, η πρόσθετη επιβάρυνση που επιφέρει στη μηχανή αυτή η εκτέλεση της συγκεκριμένης διεργασίας είναι $t_{i,j} = p_j/s_i$. Αν A είναι το σύνολο όλων των δυνατών αναθέσεων των διεργασιών στις μηχανές, τότε για οποιαδήποτε ανάθεση $\alpha \in A$, θεωρούμε ότι το στοιχείο $\alpha_{i,j}$ είναι μια ενδεικτική μεταβλητή που παίρνει την τιμή 1 αν και μόνο αν η διεργασία j ανατίθεται (βάσει του α) στη μηχανή i, διαφορετικά ισούται με 0. Για οποιαδήποτε ανάθεση $\alpha \in A$, κάθε μηχανή i

επιβαρύνεται με κάποιο **κόστος εκτέλεσης**, που ισούται με το συνολικό χρόνο εκτέλεσης όλων των διεργασιών που ανατίθενται σε αυτήν:

$$\mathbf{cost_i(t_i,\alpha) = cost_i(1/s_i,\alpha) = (1/\ s_i) \cdot [\Sigma_{j \in N}\ \alpha_{ij} \cdot p_j\ ]}$$

Το μέγιστο κόστος εκτέλεσης (για τη συγκεκριμένη ανάθεση α) καλείται **χρόνος περάτωσης** (makespan) της ανάθεσης: $\text{makespan}(t,\alpha) = \max_{i \in N}\{\text{cost}_i(t_i,\alpha)\}$. Στόχος του προβλήματος είναι η εύρεση μιας ανάθεσης α^* που εξασφαλίζει τον ελάχιστο χρόνο περάτωσης: $\alpha^* \in \arg\min_{\alpha \in A}\{\text{makespan}(t,\alpha)\} = \arg\min_{\alpha \in A} \max_{i \in N}\{\text{cost}_i(t_i,\alpha)\}$.

Το συγκεκριμένο πρόβλημα είναι NP-hard.

Στην παρούσα εργασία θεωρούμε κάθε μηχανή ως έναν «εγωιστικά σκεπτόμενο» παίκτη, ο οποίος στοχεύει στην ελαχιστοποίηση του δικού του κόστους εκτέλεσης. Η ταχύτητα s_i της μηχανής, προσδιορίζει τον **τύπο της, $t_i = 1/s_i$,** ο οποίος είναι γνωστός μόνο σε αυτήν. Αντίθετα, θεωρούμε ότι η πληροφορία που αφορά (όλες) τις διεργασίες (δηλαδή, τους φόρτους εξυπηρέτησής τους, p_j), είναι δημόσια πληροφορία. Πρέπει να επισημανθεί ότι, η κρυφή πληροφορία t_i του παίκτη είναι ένας θετικός πραγματικός αριθμός, δηλαδή οι παίκτες έχουν μόνο μια παράμετρο (one-parameter agents).

Κάθε παίκτης (μηχανή) θεωρείται ότι έχει ως συνάρτηση αποτίμησης (valuation function) για οποιαδήποτε ανάθεση $\alpha \in A$, το αντίθετο του κόστους εκτέλεσης με το οποίο επιβαρύνεται: $v_i(t_i\ ,\ \alpha) = -\text{cost}_i(t_i,\alpha)$. Προκειμένου να ελαχιστοποιήσει το δικό του κόστος εκτέλεσης, ενδέχεται ακόμη και να δώσει στο σύστημα *ψευδή στοιχεία* ως προς τον δικό του τύπο (δηλαδή, τη δική του ταχύτητα εξυπηρέτησης). Ως αντιστάθμισμα, το σύστημα (που αποφασίζει την τελική ανάθεση των διεργασιών στις μηχανές) χρησιμοποιεί κάποιες αποζημιώσεις (πληρωμές) προς τις μηχανές, για την προσπάθεια που καταβάλλουν.

Στόχος είναι η επιλογή της κατάλληλης ανάθεσης, που σε συνδυασμό με τις κατάλληλες πληρωμές, (i) αναγκάζει τις μηχανές να δηλώσουν τον πραγματικό τους τύπο (ταχύτητα εκτέλεσης), και (ii) επιτυγχάνει μια επιλογή ανάθεσης που κατά το δυνατόν είναι όσο πιο κοντά γίνεται στη βέλτιστη ανάθεση (σε σχέση με τους φόρτους επεξεργασίας των διεργασιών και τις *πραγματικές* ταχύτητες εκτέλεσης των μηχανών).

Το σύστημα λοιπόν, υιοθετεί έναν μηχανισμό (f, P), προκειμένου να επιλέξει μια συγκεκριμένη ανάθεση των διεργασιών στις μηχανές, εκτελώντας το εξής πρωτόκολλο:

1. Αρχικά το σύστημα ζητά από κάθε παίκτη i να κάνει τη δική του *δήλωση τύπου,* $b_i > 0$, η οποία δεν είναι κατ' ανάγκη φιλαλήθης.

2. Με τη βοήθεια της συνάρτησης $f : (R_{>0})^m \rightarrow A$, επιλέγει μια συγκεκριμένη ανάθεση $\alpha \in A$, ως συνάρτηση του διανύσματος δηλώσεων τύπων, b, που παρέχουν οι μηχανές.

3. Με τη βοήθεια μιας (για κάθε παίκτη i) συνάρτησης κοστολόγησης, $P_i : R_{>0}{}^m \rightarrow R$, υπολογίζεται η αποζημίωση που δίνεται στον παίκτη (μηχανή) i, ως συνάρτηση των δηλώσεων τύπων b που έχουν κάνει οι παίκτες.

4. Οι μηχανές γνωρίζουν εκ των προτέρων τον αλγόριθμο ανάθεσης f, καθώς και το σχήμα κοστολόγησης $P = (P_i)_{i \in N}$, και (θεωρούμε ότι) θέλουν να επιλέξουν την στρατηγική (δηλαδή, να κάνουν εκείνη την δήλωση τύπου) που θα μεγιστοποιήσει την προσωπική τους ωφέλεια (την αποζημίωση που θα πάρουν μείον το κόστος που προκαλείται από την εκτέλεση των διεργασιών που τους έχουν ανατεθεί). Δηλαδή: $\forall b_{-i}, b_i \in \arg\max \{ U_i(t_i , (b_{-i}, b_i))\}$

$$\forall\ \mathbf{b = (b_i) \in (R_{>0})^m,\ U_i(t_i , b) = P_i(b) - v_i\,(t_i, f(b))}$$

Ο στόχος του σχεδιαστή του μηχανισμού είναι να βρει έναν αλγόριθμο ανάθεσης και ένα σχήμα κοστολόγησης, πολυωνυμικού χρόνου, που αφενός θα βελτιστοποιήσει το makespan, σε σχέση πάντα με τις δηλώσεις των μηχανών $b = (b_i)$, και αφετέρου θα υποκινήσει τους λελογισμένους παίκτες να είναι φιλαλήθεις. Αφού το πρόβλημα $Q||C_{max}$ είναι NP-hard, θα χρησιμοποιηθεί ένας προσεγγιστικός[5] αλγόριθμος για την εύρεση της ανάθεσης.

Πιο συγκεκριμένα, η ανάθεση $\alpha \in A$ του αλγορίθμου θα αναθέτει κάποιο φορτίο (load/work) $w_i(\alpha) = \Sigma_{j \in N} \alpha_{ij} \cdot p_j$ σε κάθε παίκτη i, και θα ισχύει ότι $v_i(t_i,\alpha) = - t_i \cdot w_i(\alpha) = -w_i(\alpha) / s_i$. Έτσι, η κρυφή πληροφορία t_i κάθε παίκτη i μετράει το κόστος που του αναλογεί ανά μονάδα εργασίας.

4.2. Μορφή Συναρτήσεων Κοστολόγησης για Φιλαλήθεις Μηχανισμούς

Ενδιαφερόμαστε να σχεδιάσουμε μηχανισμούς που η «φιλαλήθεια» θα είναι κυρίαρχη στρατηγική για κάθε παίκτη. Αυτό σημαίνει ότι ο παίκτης i μεγιστοποιεί τα κέρδη του κάθε φορά που δηλώνει την πραγματική του ταχύτητα, $t_i = 1/s_i$, ανεξάρτητα με τις δηλώσεις τύπων των άλλων παικτών (b_{-i}):

$$\mathbf{U_i(t_i, (b_{-i}, t_i)) \geq U_i (t_i, (b_{-i}, b_i))}$$

Για να παράγουμε έναν τύπο για τις αποζημιώσεις P_i (συναρτήσεις κοστολόγησης), καθορίζουμε έναν παίκτη i και θεωρούμε ότι:

1. Ο μηχανισμός M = (f, P) είναι φιλαλήθης

[5] Ένας αλγόριθμος που υπολογίζει, για οποιοδήποτε στιγμιότυπο ενός προβλήματος Π, μια έξοδο που η τιμή της είναι σίγουρο ότι θα αποκλίνει κατά έναν παράγοντα ρ από την τιμή της εξόδου του βέλτιστου αλγορίθμου (για το ίδιο στιγμιότυπο), ονομάζεται ρ-προσεγγιστικός αλγόριθμος.

2. Κάθε αποζημίωση P_i (b_{-i}, b_i) και φόρτος εργασίας $w_i(f(b_{-i}, b_i))$ είναι δύο φορές παραγωγίσιμα όσον αφορά το b_i, για όλες τις τιμές του b_{-i}

3. Οι δηλώσεις b_{-i} των άλλων παικτών είναι σταθερές

4. Οι αποζημιώσεις P_i, η εργασία w_i και το κέρδος U_i είναι συναρτήσεις της δήλωσης b_i του παίκτη i

Θεωρήσαμε ότι ο μηχανισμός είναι φιλαλήθης, δηλαδή, το κέρδος του παίκτη i μεγιστοποιείται όταν ο παίκτης δηλώσει την πραγματική του τιμή t_i. Επομένως, στο σημείο t_i η πρώτη παράγωγος της συνάρτησης κέρδους πρέπει να είναι μηδενική και η δεύτερη παράγωγος πρέπει να είναι μη θετική:

Η **1η Παράγωγος** δίνει $(U_i)' = 0 \Rightarrow \left[\frac{dP_i(b_i)}{db_i} - t_i \frac{dw_i(b_i)}{db_i} \right]_{b_i = t_i} = 0$ (σχέση 4.1)

για όλες τις τιμές t_i.

Ολοκληρώνοντας κατά μέλη παίρνουμε $P_i(b_i) - P_i(0) - b_i\, w_i(b_i) + \int_0^{b_i} w_i(u)du = 0$,

δηλαδή $P_i(b_i) = P_i(0) + b_i\, w_i(b_i) - \int_0^{b_i} w_i(u)du$ (σχέση 4.2)

Η **2η Παράγωγος** δίνει $P_i''(t_i) - t_i w''(t_i) \leq 0$.

Από την σχ. 4.1, έχουμε

$$P_i'(b_i) = t_i w'(b_i) \Rightarrow P_i''(t_i) = w'(t_i) - t_i w''(t_i)$$

Από τις παραπάνω σχέσεις έχουμε ότι η δεύτερη παράγωγος δίνει $w'(t_i) \leq 0$.

Επομένως, για να είναι φιλαλήθης ο μηχανισμός, θα πρέπει να έχουμε φθίνουσες καμπύλες εργασίας (work curves) w_i, και οι αποζημιώσεις να έχουν τη μορφή της σχέσης 4.2.

4.3. Ορισμός της Ιδιότητας ¨Φθίνουσας Καμπύλης Εργασίας¨

ΟΡΙΣΜΟΣ 4.1 [Καμπύλη Εργασίας]: Με σταθερές τις δηλώσεις των άλλων παικτών $\mathbf{b}_{-i}$ και αφού έχουμε θεωρήσει την ύπαρξη κυρίαρχων στρατηγικών, η συνάρτηση w_i ($\mathbf{b}_{-i}$, b_i) εξαρτάται από μια μόνο μεταβλητή την b_i. Αυτή η συνάρτηση ονομάζεται καμπύλη εργασίας του παίκτη i. Η συνάρτηση αποτελέσματος f είναι φθίνουσα αν κάθε μια από τις σχετιζόμενες καμπύλες εργασίας είναι φθίνουσα (π.χ. η w_i (b_{-i}, b_i) είναι φθίνουσα συνάρτηση του b_i, για κάθε i και b_i).

Ουσιαστικά, ο σχεδιασμός ενός φιλαλήθη μηχανισμού ανάγεται στο σχεδιασμό αλγορίθμων ανάθεσης με φθίνουσες καμπύλες εργασίας [ArTa01].

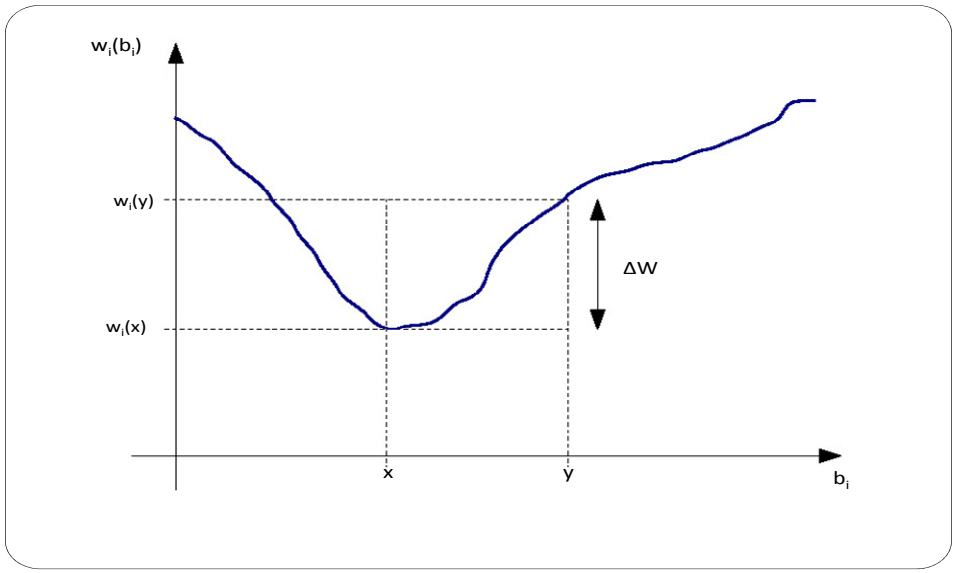
ΘΕΩΡΗΜΑ 4.2 [Μορφή Συναρτήσεων Κοστολόγησης]: Για οποιαδήποτε συνάρτηση ανάθεσης f(b) υπάρχει αλγόριθμος κοστολόγησης P(b) τέτοιος ώστε ο μηχανισμός να είναι φιλαλήθης αν και μόνο αν η συνάρτηση w_i είναι φθίνουσα ως προς το b_i για οποιαδήποτε b_{-i}. Σε αυτήν την περίπτωση, ο μηχανισμός είναι φιλαλήθης αν και μόνο αν οι αποζημιώσεις P_i (b_{-i}, b_i) είναι της μορφής:

$P_i(b_{-i}, b_i) = h_i(b_{-i}) + b_i w_i(b_{-i}, b_i) - \int_0^{b_i} w_i(b_{-i}, u)\,du$	Εξ. 4.3

όπου h_i τυχαίες συναρτήσεις

4.4. Απόδειξη Αναγκαιότητας της ¨Φθίνουσας Καμπύλης Εργασίας¨

Αρχικά, για την απόδειξη του θεωρήματος 3.2, θα εξάγουμε τις αναγκαίες συνθήκες για να έχουμε φιλαλήθεις παίκτες, οι οποίες παρακάτω αποδεικνύεται ότι είναι και ικανές.



Σχήμα 4.1: ¨Γιατί η καμπύλη εργασίας δε μπορεί να είναι αύξουσα¨

▪ **Έστω ότι η πραγματική τιμή του παίκτη i είναι $t_i = y$, θα πρέπει να ισχύει ότι**

$$\mathbf{U(y) \geq U(x) \Rightarrow U(y) - U(x) \geq 0} \qquad \text{(σχέση 4.4)}$$

Δηλαδή,

$U(y) - U(x) = P(y) - v(y, f(y)) - P(x) + v(y, f(x))$

$$= h + yw(y) - \int_0^y w(u)du - yw(y) - h - xw(x) + \int_0^x w(u)du + yw(x)$$

$$= \int_0^x w(u)du - \int_0^y w(u)du + (y-x)\,w(x) = -\int_x^y w(u)du + (y-x)\,w(x)$$

$$-\int_x^y w(u)du + (y-x)\,w(x) \leq -(y-x)\,w(x) + (y-x)\,w(x)$$

$\Rightarrow$(y–x) [$w(x) - w(x)$] = 0

Η σχέση 4.4 δεν ικανοποιείται αφού η ποσότητα $-\int_x^y w(u)du + (y-x)\, w(x)$ *είναι σίγουρα μικρότερη του μηδενός***.**

▪ **Έστω ότι η πραγματική τιμή του παίκτη i είναι $t_i = x$, θα πρέπει να ισχύει ότι**

U(x) ≥ U(y) ⇒ U(x) − U(y) ≥ 0 (σχέση 4.5)

U(x) − U(y) = P(x) – v (x, f(x)) – P (y) + v (x, f(y))

$= h + xw(x) - \int_0^x w(u)du - xw(x) - h - yw(y) + \int_0^y w(u)du + xw(y)$

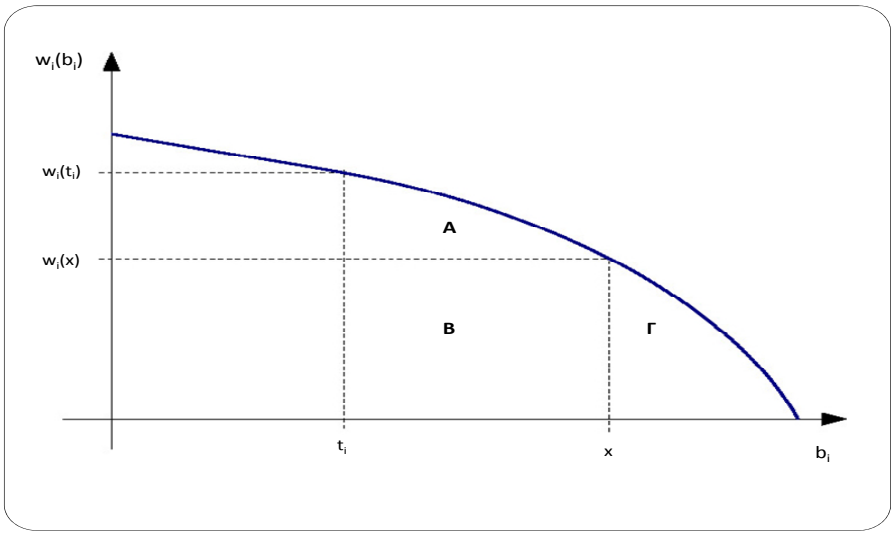
$= -\int_0^x w(u)du - yw(y) + \int_0^y w(u)du + xw(y) = \int_x^y w(u)du$ -(y-x) $w(y)$

$\int_x^y w(u)du - (y-x)\, w(y) \le (y-x)\, w(y) - (y-x)\, w(y) \Rightarrow (y-x)\,[\, w(y) - w(y)\,] = 0$

Άρα, η σχέση 4.5 δεν ικανοποιείται αφού η ποσότητα $\int_x^y w(u)du - (y-x)\, w(y)$ *είναι σίγουρα μικρότερη του μηδενός.*

Η καμπύλη εργασίας θα έπρεπε να φθίνει μονοτονικά στο σχήμα 4.

Με τη βοήθεια των σχημάτων 4.2α και 4.2β, θα δείξουμε ότι η φθίνουσα μονοτονικότητα της καμπύλης εργασίας, εξασφαλίζει φιλαλήθεις μηχανισμούς.



Σχήμα 4.2α: ¨Ο παίκτης i δεν κερδίζει ποτέ αν «μπλοφάρει»¨

Έστω ότι t η πραγματική τιμή του παίκτη i. Ο παίκτης έχει τη δυνατότητα να δηλώσει μεγαλύτερη τιμή x > t με w(x) < w(t) (βλ. σχήμα 4.2α). Θα πρέπει να δείξουμε ότι ο παίκτης μεγιστοποιεί το κέρδος του μόνο όταν δηλώσει την πραγματική του τιμή t. Δηλαδή,

$$\mathbf{U(t) - U(x) \geq 0} \qquad \text{(σχέση 4.6)}$$

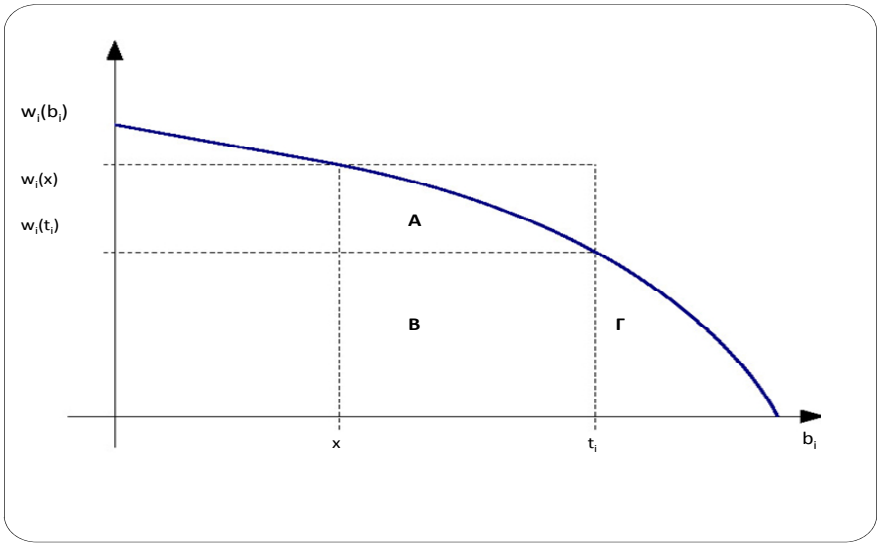
$U(t) - U(x) = P(t) - v(t, f(t)) - P(x) + v(t, f(x))$

$$= h + tw(t) - \int_0^t w(u)du - tw(t) - h - xw(x) + \int_0^x w(u)du + tw(x)$$

$$= \int_t^x w(u)du - xw(x) + tw(x) \geq (x-t)\ w(x) + (t-x)\, w(x)$$

$$= (t-x)[\, w(x) - w(x)\,] = 0$$

Επομένως, η ποσότητα $\int_t^x w(u)du - xw(x) + tw(x)$ *είναι σίγουρα μεγαλύτερη από το μηδέν και άρα ισχύει πάντα η σχέση 4.6.*



Σχήμα 4.2β: ¨Ο παίκτης i δεν κερδίζει ποτέ αν «μπλοφάρει»¨

Έστω ότι t η πραγματική τιμή του παίκτη i. Ο παίκτης έχει τη δυνατότητα να δηλώσει μικρότερη τιμή y < t με w(y) < w(t) (βλ. σχήμα 4.2β). Θα πρέπει να δείξουμε ότι ο παίκτης μεγιστοποιεί το κέρδος του μόνο όταν δηλώσει την πραγματική του τιμή t. Δηλαδή,

$$\mathbf{U(t) - U(y) \geq 0} \qquad \text{(σχέση 4.7)}$$

U(t) - U(y) = P(t) – v (t, f(t)) - P (y) + v (t, f(y)) =

$$= h + tw(t) - \int_0^t w(u)du - tw(t) - h - yw(y) + \int_0^y w(u)du + tw(y) =$$

$$= -\int_y^t w(u)du - yw(y) + tw(y) \geq -(t-y)\ w(y) + (t-y)\, w(y) =$$

$$= (t-y)[\, w(y) - w(y)\,] = 0$$

Επομένως, η ποσότητα $-\int_y^t w(u)du - yw(y) + tw(y)$ *είναι σίγουρα μεγαλύτερη από το μηδέν και άρα ισχύει πάντα η σχέση 4.7.*

Δείξαμε ότι η ιδιότητα της Φθίνουσας Καμπύλης Εργασίας (DWC), για οποιαδήποτε συνάρτηση ανάθεσης f και δεδομένου του σχήματος κοστολόγησης της σχέσης 4.3 είναι αναγκαία συνθήκη για να έχουμε φιλαλήθεις μηχανισμούς.

4.5. Απόδειξη Ικανότητας της ¨Φθίνουσας Καμπύλης Εργασίας¨

Πρέπει να εξασφαλιστεί ότι μια συνάρτηση ανάθεσης f είναι υλοποιήσιμη (implementable), δηλαδή, ότι πράγματι υπάρχουν οι κατάλληλες συναρτήσεις κοστολόγησης, ($P_κ$), που αποδεικνύουν την φιλαλήθεια του αντίστοιχου μηχανισμού, (f, P).

Έστω ότι έχουμε έναν φιλαλήθη μηχανισμό (f, P) κι έστω ότι η συνάρτηση ωφέλειας (το κέρδος) ενός παίκτη όταν δηλώνει τον πραγματικό του τύπο $t \geq 0$ είναι:

$$\mathbf{U(t) = P(t) – v\,(t, f(t))}$$

Αφού ο μηχανισμός είναι φιλαλήθης, ο παίκτης μεγιστοποιεί την ωφέλειά του όταν δηλώνει τον πραγματικό του τύπο. Επομένως, για όλα τα $b \geq 0$, ισχύει:

$U(t) \geq P(b) – v\,(t, f(b)) \Rightarrow U(t) \geq P(b) – tw(b) \Rightarrow$

$U(t) \geq [P(b) – bw(b)] + bw(b) - tw(b) \Rightarrow$

$U(t) \geq U(b) + bw(b) - tw(b) \Rightarrow$

$U(t) \geq U(b) + (t – b)[- w(b)]$ (σχέση 4.8)

Η σχέση 4.8 απεικονίζει την ιδιότητας της φιλαλήθειας.

Χρησιμοποιώντας τη σχέση 4.2: $P(t) = P(0) + t\,w(t) - \int_0^t w(u)du$ και τον ορισμό της συνάρτησης ωφέλειας: $U(b) = P(b) – v\,(t, f(b))$

Έχουμε P(t) – v (t, f(t))≥ P(b) – v (t, f(b)) $\Rightarrow$

$$P(0) + t\,w(t) - \int_0^t w(u)du \text{ - tw(t)} \geq P(0) + b\,w(b) - \int_0^b w(u)du \text{ - tw(b)} \Rightarrow$$

$$\int_0^b w(u)du - \int_0^t w(u)du \geq b\,w(b) - \text{tw(b)} \Rightarrow$$

$$\int_b^t w(u)du \leq (t-b)w(b)$$

Αφού η συνάρτηση $w(b)$ είναι φθίνουσα επιβεβαιώνουμε την ικανότητα της ιδιότητας DWC και καταλήγουμε στο ότι ένας μηχανισμός (f, P), για έναν παίκτη, με τύπο t και συνάρτηση ωφέλειας U(t) = P(t) – v (t, f(t)), είναι φιλαλήθης αν και μόνο αν $w(b)$ είναι φθίνουσα συνάρτηση του b, και $\text{P(b)} = P(0) + b\,w(b) - \int_0^b w(u)du$.

4.6. Οικειοθελής Συμμετοχή (Voluntary Participation)[6]

Από το σχήμα κοστολόγησης (αποζημιώσεων), που δίνεται στο θεώρημα 4.2. (σχέση 4.3), φαίνεται ότι οι μόνες τροποποιήσεις που μπορούν να γίνουν στο σχήμα αφορούν τους σταθερούς όρους $h_i(b_{-i})$.

Αν θέσουμε όλους αυτούς τους όρους ίσους με το μηδέν $h_i(b_{-i}) = 0$, τότε το κόστος ενός φιλαλήθη παίκτη θα είναι $t_i w(t_i)$, που αναιρεί το δεύτερο (θετικό) όρο του σχήματος αποζημιώσεων. Επομένως, ο παίκτης τελικά θα έχει μια απώλεια που θα ισούται με την περιοχή κάτω από την καμπύλη εργασίας και μεταξύ του μηδενός και του t_i (τρίτος όρος σχήματος αποζημιώσεων, $-\int_0^{b_i} w_i(b_{-i},u)du$. Αφού, με αυτό το σχήμα, οι παίκτες δεν πρόκειται να κερδίσουν, το πιθανότερο είναι ότι δε θα συμμετέχουν στο παιχνίδι.

[6] Ο Όρος Voluntary Participation είναι επίσης γνωστός και ως Individual Rationality

ΟΡΙΣΜΟΣ 4.3 [Ιδιότητα Οικειοθελούς Συμμετοχής]: Ένας μηχανισμός ικανοποιεί την ιδιότητα οικειοθελούς συμμετοχής, αν οι παίκτες, που δηλώνουν τις πραγματικές τους τιμές, έχουν πάντα κέρδος μεγαλύτερο ή ίσο του μηδενός. Δηλαδή, $U_i(t_i, (b_{-i}, t_i)) \geq 0$, για κάθε παίκτη i, για τις πραγματικές τιμές t_i, και τις δηλώσεις b_{-i} των άλλων παικτών.

Προκειμένου να κατασκευάσουμε μηχανισμούς που να ικανοποιούν την ιδιότητα της οικειοθελούς συμμετοχής θα πρέπει να ορίσουμε τη σταθερά $h_i(b_{-i})$ έτσι ώστε να είναι μεγαλύτερη από την περιοχή κάτω από την καμπύλη εργασίας και αριστερά του t_i, **ανεξαρτήτως** της τιμής του t_i. Αν ολόκληρη η περιοχή κάτω από την καμπύλη εργασίας είναι άπειρη τότε δεν υπάρχει τέτοια σταθερά. Αν η περιοχή είναι πεπερασμένη τότε μπορούμε να ορίσουμε τη σταθερά $h_i(b_{-i})$ ίση με αυτήν την περιοχή. Σε αυτήν την περίπτωση, ο φιλαλήθης παίκτης θα έχει ένα κέρδος που θα ισούται με την περιοχή κάτω από την καμπύλη εργασίας και προς τα δεξιά του t_i. Στο σχήμα 2, το κέρδος θα ισούται με την περιοχή Α+Β+Γ.

ΘΕΩΡΗΜΑ 4.4 [Οικειοθελής Συμμετοχή και Συνάρτηση Κοστολόγησης]: Δεδομένης μιας μονοτονικά φθίνουσας συνάρτησης ανάθεσης f(b), μπορούμε να επιλέξουμε ένα σχήμα κοστολόγησης (αποζημιώσεων), που να ικανοποιεί τις ιδιότητες της φιλαλήθειας και της οικειοθελούς συμμετοχής, αν και μόνο αν $\int_0^\infty w_i(b_{-i}, u)du < \infty$, για όλα τα i και b_{-i}.

Σε αυτήν την περίπτωση μπορούμε να πάρουμε το παρακάτω σχήμα κοστολόγησης:

$P_i(b_{-i}, b_i) = b_i w_i(b_{-i}, b_i) + \int_{b_i}^\infty w_i(b_{-i}, u)du$	Εξ. 4.9

4.7. Μηχανισμός Ανάθεσης Διεργασιών Μη Πολυωνυμικού Χρόνου

Γενικότερα, ο στόχος είναι να καταλήξουμε σε μια ανάθεση f(b) που θα αναθέτει μειούμενη ποσότητα εργασίας σε έναν παίκτη i, του οποίου το δηλωθέν κόστος ανά μονάδα εργασίας αυξάνεται.

Ταξινομούμε κατά αύξουσα σειρά τους χρόνους επεξεργασίας των διεργασιών $p_1 \geq \ldots \geq p_n$. Κάθε μηχανή i γνωρίζουμε ότι χρειάζεται p_j/s_i χρονικές μονάδες για να εκτελέσει την διεργασία j. Το συνολικό φορτίο στη μηχανή είναι $w_i(b) = \Sigma_j (f(\mathbf{b}))_{ij} \cdot p_j$. Κάθε μηχανή επιφέρει ένα κόστος ανάλογο του χρόνου που χρειάζεται για να επεξεργαστεί τις διεργασίες που της έχουν ανατεθεί.

ΟΡΙΣΜΟΣ 4.5 [Λεξικογραφική Διάταξη]: Ένα διάνυσμα $(w_1,\ldots,w_m)$ είναι μικρότερο λεξικογραφικά από ένα άλλο διάνυσμα $(w_1',\ldots,w_m')$, αν για κάποιο i, $w_i < w_i'$ και $w_k = w_k'$ για όλα τα $k < i$.

ΠΡΟΤΑΣΗ 4.6 [Μη πολυωνυμικός Φιλαλήθης Μηχανισμός]: Υπάρχει ένας φιλαλήθης μηχανισμός (όχι πολυωνυμικός) που εξάγει μια βέλτιστη λύση για το πρόβλημα $Q||C_{max}$ και ικανοποιεί την ιδιότητα οικειοθελούς συμμετοχής.

Απόδειξη: Ανάμεσα στις βέλτιστες αναθέσεις διεργασιών, ο αλγόριθμος επιλέγει εκείνη όπου το διάνυσμα $(w_1,\ldots,w_m)$ είναι λεξικογραφικά το μικρότερο. Όσο η μηχανή αυξάνει την τιμή b_i που δηλώνει (ουσιαστικά μειώνει την ταχύτητά της s_i), δε θα υπάρξει αλλαγή στην ανάθεση διεργασιών, εκτός κι αν αυτή η μηχανή αποτελέσει bottleneck. Σε αυτήν την περίπτωση, αυξάνει τη δήλωσή της b_i με αποτέλεσμα να παίρνει λιγότερη εργασία. Έτσι, η συνάρτηση ανάθεσης είναι φθίνουσα, και σύμφωνα με το θεώρημα 4.2 υπάρχει σχήμα αποζημιώσεων που δίνεται από την εξίσωση 4.3.

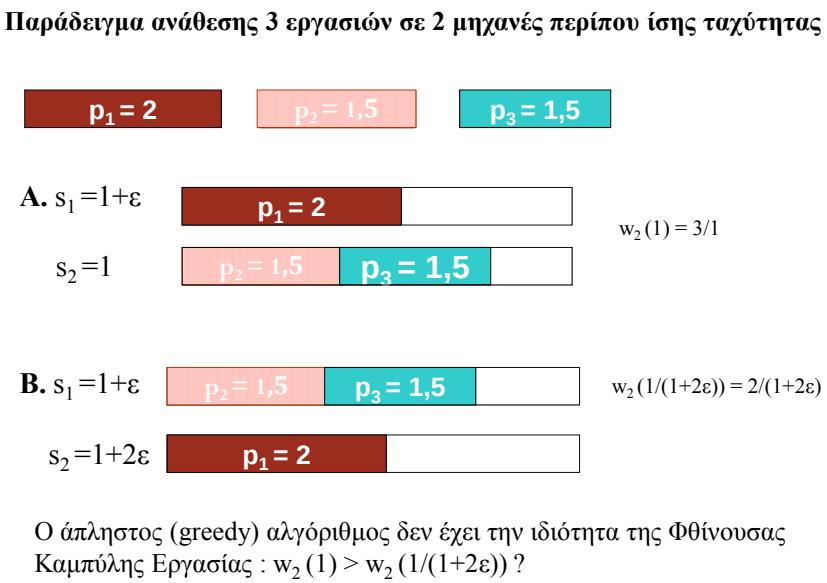
Σε γενικές γραμμές, η ποσότητα $w_i(b_{-i},\cdot)$ είναι σταθερή, εκτός από τα σημεία όπου η μηχανή i υπερφορτώνεται και γίνεται bottleneck. Άρα οι αποζημιώσεις υπολογίζονται σχετικά εύκολα. Επιπλέον, σε μια αρκετά αργή μηχανή δεν ανατίθεται καμία διεργασία

κι επομένως από το θεώρημα 4.4, οι αποζημιώσεις μπορούν να επιλεχθούν έτσι ώστε να ικανοποιούν την ιδιότητα οικειοθελούς συμμετοχής.

4.8. Μηχανισμός Ανάθεσης Διεργασιών Πολυωνυμικού Χρόνου

Είναι σαφές ότι, ο συνολικός φόρτος εργασίας μιας μηχανής i μεταβάλλεται καθώς μεταβάλλεται η ταχύτητά της, δηλαδή τα b_i που δηλώνει. Επομένως, ο προσεγγιστικός αλγόριθμος που θα χρησιμοποιηθεί θα πρέπει να αναθέτει διεργασίες στις μηχανές με φθίνοντα τρόπο, δηλαδή έτσι ώστε καθώς αυξάνεται το b_i, και άρα μειώνεται η ταχύτητά της μηχανής, να μειώνεται και ο φόρτος εργασίας που ανατίθεται στη μηχανή.

Κάτι τέτοιο δεν επιτυγχάνει, για παράδειγμα, ο **άπληστος (greedy)** αλγόριθμος. Με ένα παράδειγμα θα μελετήσουμε τον άπληστο κανόνα ανάθεσης, σύμφωνα με τον οποίο οι διεργασίες ταξινομούνται κατά φθίνουσα σειρά όσον αφορά τους χρόνους επεξεργασίας τους, που είναι και η σειρά με την οποία θα ανατεθούν στις μηχανές. Πριν την ανάθεση μιας διεργασίας ελέγχεται ο φόρτος εργασίας όλων των μηχανών και η εργασία ανατίθεται στη μηχανή με το μικρότερο φόρτο εργασίας. Έστω ότι έχουμε τρεις διεργασίες με χρόνους επεξεργασίας $p_1 = 2$, $p_2 = p_3 = 1{,}5$ και δύο μηχανές με ταχύτητες $s_1 = 1 + \varepsilon$ και $s_2 = 1$.



Σχήμα 4.3: ¨Μελέτη Άπληστου Κανόνα Ανάθεσης¨

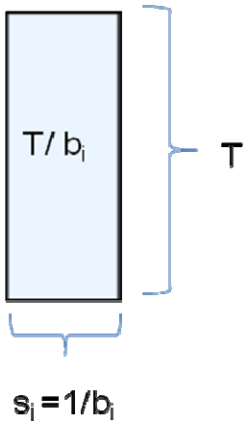
Αρχικά (στην περίπτωση Α) η μηχανή 2 δηλώνει τύπο ελάχιστα μεγαλύτερο από τη μηχανή 1, άρα έχει ταχύτητα ελάχιστα μικρότερη από τη μηχανή 1, επομένως η μεγαλύτερη εργασία θα ανατεθεί στη μηχανή 1 και οι 2 μικρότερες στη μηχανή 2. Ο φόρτος εργασίας της μηχανής 2 για τον τύπο που ισούται με 1 είναι ίσος με 3.

Στη συνέχεια (στην περίπτωση Β) η μηχανή 2 δηλώνει τύπο ελάχιστα μικρότερο από τη μηχανή 1, άρα έχει ταχύτητα ελάχιστα μεγαλύτερη από τη μηχανή 1, κι επομένως θα αναλάβει την εκτέλεση της μεγαλύτερης εργασίας ενώ οι 2 μικρότερες θα ανατεθούν στη μηχανή 1. Ο φόρτος εργασίας της μηχανής 2 για τον τύπο που ισούται με 1 προς 1+2ε είναι ίσος με 2 προς 1+2ε.

Βλέπουμε δηλαδή ότι ενώ η μηχανή δύο μειώνει τον τύπο της από 1 σε 1/ (1+2 ε) αντί να έχουμε αύξηση του φόρτου εργασίας της έχουμε μείωση από 3 σε 2/ (1+2 ε). Άρα ο άπληστος αλγόριθμος ανάθεσης δεν εξασφαλίζει φιλαλήθεια.

Παρακάτω περιγράφονται τα βήματα που οδηγούν στην κατασκευή ενός πολυωνυμικού χρόνου, 3- προσεγγιστικού μηχανισμού για το πρόβλημα Q| |C_{max}.

Υποθέτουμε μια τιμή T για την τιμή του βέλτιστου makespan $C^*_{max,}$ και για κάθε μηχανή παίρνουμε ένα μέγεθος T/b_i, που είναι το μέγιστο φορτίο που μπορούμε να αναθέσουμε στη μηχανή εάν θέλουμε να εκτελέσει όλες τις διεργασίες της σε χρόνο το πολύ T. Σε κάθε μηχανή δημιουργούμε ένα καλάθι μεγέθους T/b_i (βλ. σχήμα 4.4).



Σχήμα 4.4: Καλάθι Μεγέθους T/b_i

<u>Μηχανισμός Κλασματικών Αναθέσεων (Fractional Assignment)</u>

Μπορούμε να επιτρέψουμε κλασματικές αναθέσεις διεργασιών στα καλάθια: διαμέριση κάθε διεργασίας j σε μέρη, δηλαδή σε υποεργασίες, συνολικού μεγέθους p_j και ανάθεση αυτών των διαμερισμένων διεργασιών στα καλάθια.

Μια κλασματική ανάθεση είναι έγκυρη:

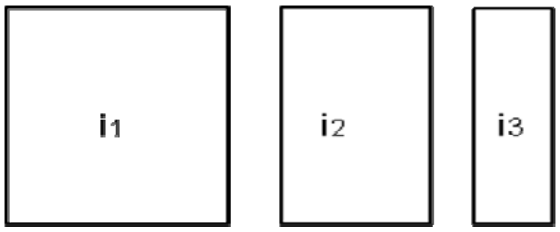
1. αν κάθε καλάθι έχει μέγεθος τουλάχιστον ίσο με το συνολικό μέγεθος όλων των υποεργασιών που ανατίθενται σε αυτό, $\sum_j x_{ij} p_j \le T / b_i$ και

2. αν το μέγεθος του καλαθιού, που αναλαμβάνει την εκτέλεση μιας υποεργασίας, είναι αρκετά μεγάλο ώστε να χωράει ολόκληρη την διεργασία $T / b_i \ge p_j$.

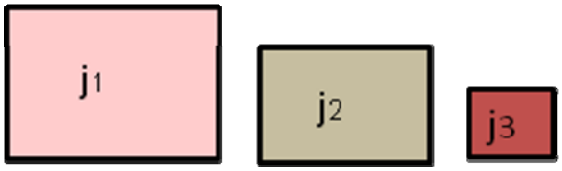
Το μικρότερο T για το οποίο υφίσταται μια έγκυρη κλασματική ανάθεση αποτελεί κάτω φράγμα για το C^*_{max}. Στόχος είναι να περιγράψουμε έναν αλγόριθμο για το T που να υπολογίζει αυτό το κάτω φράγμα.

Αν υπάρχει μια έγκυρη κλασματική ανάθεση μπορεί να βρεθεί από τον ακόλουθο άπληστο αλγόριθμο:

- Απαρίθμησε τα καλάθια από το μεγαλύτερο στο μικρότερο ($b_1 \le \ldots \le b_m$)



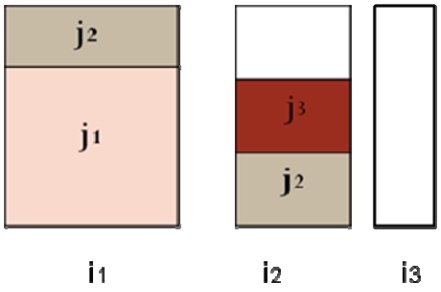
- Απαρίθμησε τις διεργασίες από τη μεγαλύτερη στη μικρότερη ($p_1 \ge \ldots \ge p_n$)



- Ανέθεσε τις διεργασίες 1, 2, …, (k-1) στο καλάθι 1, όπου k είναι η πρώτη διεργασία που θα προκαλέσει υπερχείλιση στο καλάθι

- Ανέθεσε στο καλάθι 1 ένα μέρος της διεργασίας k μεγέθους ίσου με τον ελεύθερο χώρο του καλαθιού 1

- Συνέχισε αναθέτοντας διεργασίες στο καλάθι 2, και ξεκίνησε με την ανάθεση του υπολειπόμενου μέρους της διεργασίας k



Για κάθε διεργασία j, έστω i(j) το τελευταίο καλάθι που είναι τουλάχιστον τόσου μεγέθους όσο είναι η διεργασία j. Οι αναθέσεις που γίνονται με άπληστο τρόπο είναι έγκυρες αν και μόνο αν, για κάθε διεργασία j, η συνολική χωρητικότητα των πρώτων i(j) καλαθιών είναι τουλάχιστον ίση με το συνολικό μέγεθος όλων των πρώτων j διεργασιών.

Έτσι, αν η ανάθεση είναι έγκυρη και i είναι το τελευταίο καλάθι στο οποίο η διεργασία j ανατίθεται (πλήρως ή μερικώς),

Τότε, $T \geq \max\left\{ b_i p_j, \frac{\sum_{k=1}^{j} p_k}{\sum_{l=1}^{i} \frac{1}{b_l}} \right\}$

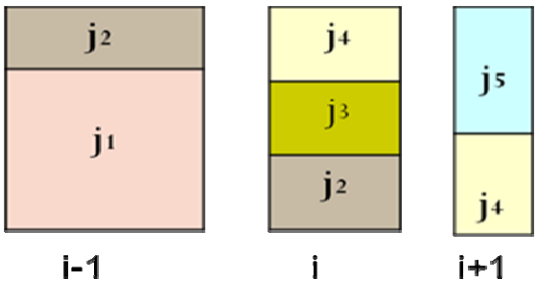
Ο πρώτος όρος στην παρένθεση εξασφαλίζει ότι η διεργασία j θα χωράει πάντα ολόκληρη στη μηχανή i, ακόμα και αν σε αυτή ανατίθεται μόνο ένα μέρος της διεργασίας j (συνθήκη εγκυρότητας 2). Ο δεύτερος όρος εξασφαλίζει ότι η συνολική

χωρητικότητα των πρώτων i(j) καλαθιών είναι τουλάχιστον ίση με το συνολικό μέγεθος όλων των πρώτων j διεργασιών (συνθήκη εγκυρότητας 1).

$$\text{Έτσι, } T_{LB} = \max_{j} \min_{i} \max \left\{ b_i p_j, \frac{\sum_{k=1}^{j} p_k}{\sum_{l=1}^{i} \frac{1}{b_l}} \right\} \qquad \text{(σχέση 4.10)}$$

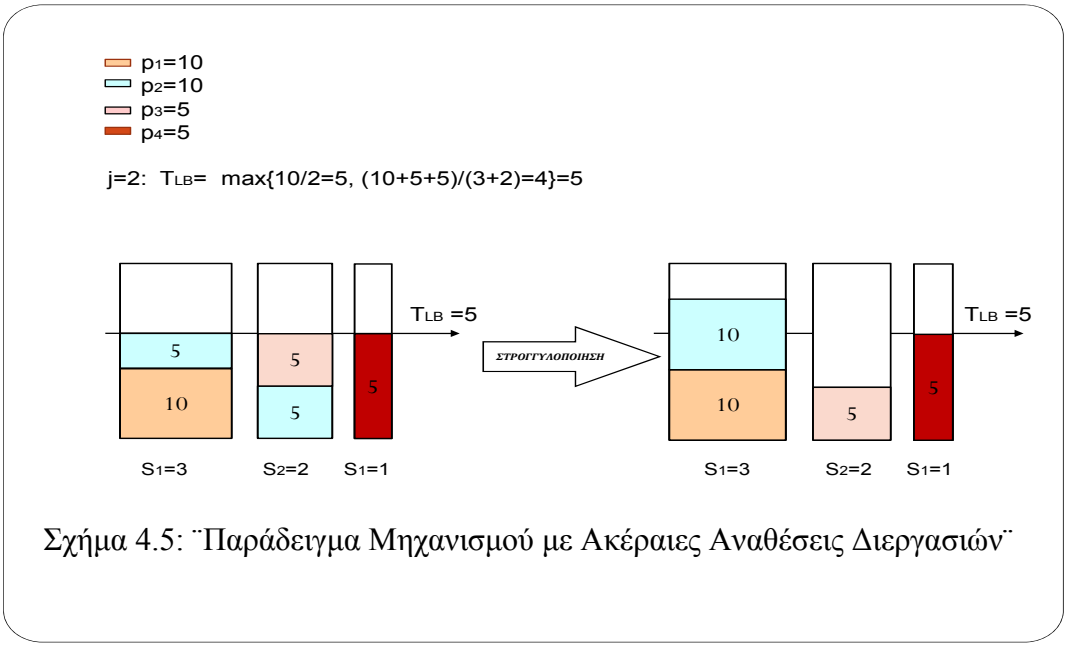
Το T_{LB} συμβολίζει το κάτω φράγμα του makespan, C^*_{max}. Βρίσκουμε το T_{LB} για την τελευταία διεργασία (το μεγαλύτερο δυνατό j) που μπορεί να ανατεθεί στις πρώτες i μηχανές (το μικρότερο δυνατό i).

ΛΗΜΜΑ 4.7: Έστω ότι το ύψος των καλαθιών ισούται με T_{LB}, ο άπληστος αλγόριθμος αποδίδει μια έγκυρη κλασματική ανάθεση, τέτοια ώστε, κάθε καλάθι να περιέχει έναν αριθμό από ολόκληρες διεργασίες και το πολύ δύο υποεργασίες, δηλαδή το πολύ δύο μέρη από δύο διαφορετικές διεργασίες.



Μηχανισμός με Ακέραιες Αναθέσεις Διεργασιών

Με δεδομένη την κλασματική ανάθεση διεργασιών που προκύπτει από τον άπληστο αλγόριθμο, ο νέος αλγόριθμος μεταφέρει την εκτέλεση κάθε διαμερισμένης διεργασίας, στην πιο γρήγορη από τις δύο μηχανές που αυτή μοιράζεται. Το φορτίο σε κάθε μηχανή, ισούται πλέον με το συνολικό μέγεθος των διεργασιών που ήταν πλήρως ανατεθειμένες στη μηχανή από την κλασματική ανάθεση (βήμα 1), συν το μέγεθος το πολύ μιας επιπλέον διεργασίας.



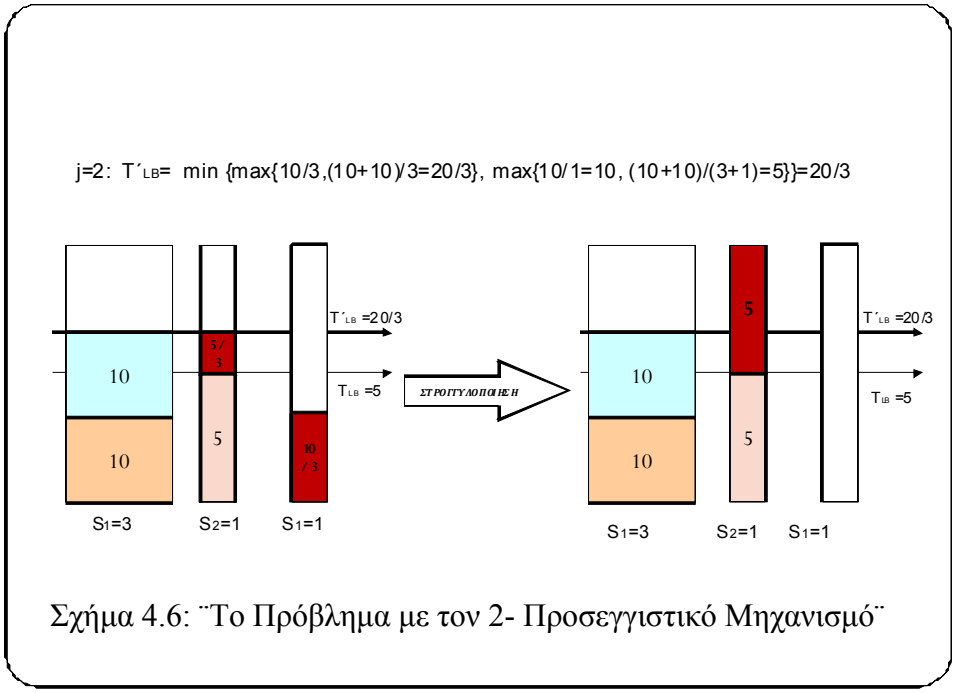
Σχήμα 4.5: ¨Παράδειγμα Μηχανισμού με Ακέραιες Αναθέσεις Διεργασιών¨

Αφού η κλασματική ανάθεση είναι έγκυρη, η στρογγυλοποιημένη (rounded) ανάθεση που προκύπτει, υπερχειλίζει κάθε καλάθι το πολύ κατά ένα παράγοντα του 2, επομένως ο μηχανισμός είναι 2- προσεγγιστικός (2- approximation):

$$T_{LB} \leq T_{LB} + b_i p_j \leq T_{LB} + T_{LB} = 2\, T_{LB} \leq 2\, C^*max$$

Ωστόσο, ***ο παραπάνω μηχανισμός δεν εξασφαλίζει φθίνουσες καμπύλες εργασίας***. Θα δούμε ένα παράδειγμα με $i > 1$ και $j < n$, και με τη διεργασία j να ολοκληρώνεται στο καλάθι i.

Στο παραπάνω παράδειγμα, παίρνουμε τη διεργασία 2 που ολοκληρώνεται στο καλάθι 2. Έστω ότι η μηχανή 2 μειώνει την ταχύτητά της από 2 σε 1 και άρα αυξάνει την τιμή *b* που δηλώνει. Από τη σχέση 4.10 βλέπουμε ότι θα αυξηθεί και το T_{LB} σε T'_{LB} ($T'_{LB} > T_{LB}$). Στο παρακάτω σχήμα φαίνεται ότι αυτό έχει ως αποτέλεσμα την ανάθεση της διεργασίας 4 στα καλάθια 2 και 3. Κατά τη διαδικασία στρογγυλοποίησης η διεργασία 4 τελικά θα ανατεθεί ολόκληρη στην πιο γρήγορη μηχανή, δηλαδή στη 2, *αυξάνοντας έτσι το φορτίο της, παρά τη μείωση της ταχύτητάς της.*



Σχήμα 4.6: ¨Το Πρόβλημα με τον 2- Προσεγγιστικό Μηχανισμό¨

Επειδή είναι δύσκολο να αντιμετωπιστεί αυτό το πρόβλημα με ντετερμινιστικό αλγόριθμο στρεφόμαστε στους πιθανοτικούς αλγόριθμους (randomized algorithms).

<u>Πιθανοτικός Μηχανισμός</u>

Δημιουργούμε συνθήκες τυχαιότητας για να εξασφαλίσουμε μονότονη καμπύλη εργασίας για κάθε μηχανή. Παίρνουμε το αποτέλεσμα του άπληστου αλγόριθμου, που είναι οι κλασματικές αναθέσεις διεργασιών σε καλάθια, και τυχαία αναθέτουμε τις διεργασίες ως εξής: η διεργασία j ανατίθεται στη μηχανή i με πιθανότητα ίση με τον

κλασματικό λόγο που δείχνει την αναλογία της διεργασίας j που ανατίθεται στη μηχανή i.

Θεωρούμε ότι ο σκοπός κάθε παίκτη είναι να μεγιστοποιήσει το *αναμενόμενο κέρδος* του. Έτσι, η φιλαλήθεια είναι κυρίαρχη στρατηγική για τον παίκτη i, αν δηλώνοντας t_i, μεγιστοποιεί το αναμενόμενο κέρδος του ανεξάρτητα με το τι έχουν δηλώσει οι υπόλοιποι παίκτες. Επίσης, ο μηχανισμός είναι φιλαλήθης αν η φιλαλήθεια είναι κυρίαρχη στρατηγική για κάθε παίκτη. ***Τώρα το w_i είναι το αναμενόμενο φορτίο του παίκτη i που ισούται με το κλασματικό φορτίο που προκύπτει από τον άπληστο αλγόριθμο.*** Με βάση το θεώρημα 4.2, ο πιθανοτικός μηχανισμός δέχεται σαν είσοδο (ντετερμινιστικά)[7] το σχήμα κοστολόγησης που εξασφαλίζει φιλαλήθεια, αν και μόνο αν το αναμενόμενο φορτίο στη μηχανή i είναι μια φθίνουσα συνάρτηση των τιμών b_i που δηλώνει ο i.

ΘΕΩΡΗΜΑ 4.8 [3- Προσεγγιστικός Πολυωνυμικός Μηχανισμός]: Η τυχαία ανάθεση που περιγράφθηκε χρησιμοποιεί ένα σχήμα κοστολόγησης που ικανοποιεί τις ιδιότητες της φιλαλήθειας και της οικειοθελούς συμμετοχής και ντετερμινιστικά εξασφαλίζει έναν πολυωνυμικού χρόνου 3- προσεγγιστικό μηχανισμό για το πρόβλημα $Q||C_{max}$.

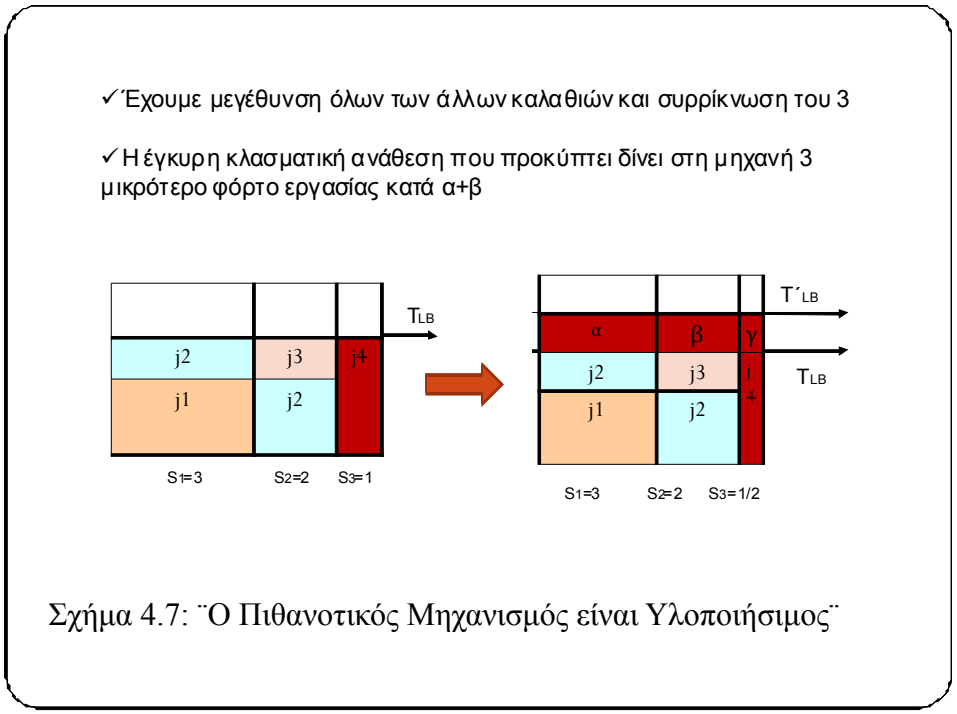
Απόδειξη: Αφού η κλασματική ανάθεση είναι έγκυρη και η στρογγυλοποίηση (με τυχαιότητα) αναθέτει σε κάθε μηχανή το πολύ δύο (2) επιπλέον διεργασίες για να εκτελέσει, κάθε καλάθι είναι το πολύ τρεις φορές γεμάτο. Έτσι, η ανάθεση που προκύπτει είναι 3- προσεγγιστική ανεξάρτητα με τις τυχαίες επιλογές που έγιναν:

$$\mathbf{T_{LB} \le T_{LB} + b_i p_j + b_i p_j \le T_{LB} + T_{LB} + T_{LB} = 3\ T_{LB} \le 3\ C^*max}$$

[7] Θα μπορούσαμε να επιλέξουμε τις αποζημιώσεις να είναι τυχαίες μεταβλητές που η αναμενόμενη τιμή τους να δίνεται από τη σχέση 4.3.

Τώρα θα δείξουμε ότι το αναμενόμενο φορτίο σε κάθε μηχανή i μειώνεται όσο ο παίκτης i δηλώνει μεγαλύτερη τιμή b_i (γίνεται δηλαδή βραδύτερη). Το αναμενόμενο φορτίο στην μηχανή i είναι ακριβώς το φορτίο που προέκυψε από τον άπληστο αλγόριθμο στην κλασματική ανάθεση: α) για γεμάτα καλάθια, το αναμενόμενο φορτίο είναι T_{LB} / b_i, β) για το (πολύ) ένα μερικώς γεμάτο καλάθι είναι το φορτίο που απέμεινε από τα γεμάτα καλάθια, και γ) για τα άδεια καλάθια είναι μηδέν.

Υποθέτουμε ότι κάποια μηχανή αλλάζει το b_i που δηλώνει με το ab_i, όπου $\alpha > 1$, γίνεται δηλαδή βραδύτερη. Σύμφωνα με τη σχέση 4.10, δημιουργείται νέο κάτω φράγμα T_{LB}'. Ισχύει ότι $T_{LB}' \geq T_{LB}$. Στο παρακάτω σχήμα, έστω ότι η μηχανή 3 μειώνει την ταχύτητά της από 1 σε ½. Αυτό έχει ως αποτέλεσμα να συρρικνώνεται το καλάθι 3 κατά ένα παράγοντα α και στη συνέχεια να μεγεθύνονται όλα τα υπόλοιπα καλάθια κατά τον ίδιο παράγοντα α, έτσι ώστε να είναι δυνατό να εκτελεστεί το μέρος της διεργασίας j4 που δε χωράει πλέον στο καλάθι 3 (α+β) από τις υπόλοιπες μηχανές, 1 και 2. Το αποτέλεσμα είναι μια έγκυρη κλασματική ανάθεση. Γενικά, η επίδραση της αύξησης της τιμής b_i της μηχανής i είναι η μεγέθυνση όλων των άλλων καλαθιών και η συρρίκνωση του i, ώστε η άπληστη κλασματική ανάθεση να δίνει στον i μικρότερο φόρτο εργασίας. Επομένως, ο πιθανοτικός μηχανισμός είναι υλοποιήσιμος.



Σχήμα 4.7: ¨Ο Πιθανοτικός Μηχανισμός είναι Υλοποιήσιμος¨

Το αναμενόμενο φορτίο $w_i(b_{-i}, b_i)$ είναι μια φθίνουσα συνάρτηση του b_i, κι επομένως από το θεώρημα 4.2 μπορούμε να κατασκευάσουμε ένα σχήμα κοστολόγησης που εξασφαλίζει φιλαλήθεια. Αν οι μηχανές που δηλώνουν αρκετά μεγάλες τιμές δε λαμβάνουν κάποια διεργασία, μπορούμε να επιλέξουμε το σχήμα κοστολόγησης έτσι ώστε να ικανοποιεί την ιδιότητα της οικειοθελούς συμμετοχής (θεώρημα 4.4).

Για να υπολογίσουμε τα κόστη/ αποζημιώσεις, πρέπει να υπολογίσουμε τη συνάρτηση $w_i(b_{-i},\square)$ και το ολοκλήρωμα $\int_{b_i}^{\infty} w_i(b_{-i}, x)dx$. Έστω ότι το $T_{LB}(x)$ είναι το κάτω φράγμα όταν ο παίκτης i δηλώνει x και οι άλλοι δηλώνουν b_{-i}. Όταν οι παίκτες δηλώνουν μικρές τιμές (γρήγορες ταχύτητες) το καλάθι i είναι γεμάτο, ενώ όταν δηλώνουν μεγάλες τιμές το φορτίο είναι μηδέν. Για το ενδιάμεσο διάστημα, το φορτίο ισούται με την διεργασία που έχει απομείνει από τα μεγαλύτερα καλάθια. Έτσι, χρειάζεται μόνο να βρούμε το $T_{LB}(x)$. Ανάλογα με το ενδιάμεσο διάστημα, αυτό είναι μια σταθερά της μορφής cx ή

της μορφής $\frac{c}{d+\frac{1}{x}}$ (όπου c και d σταθερές), πράγμα που εξαρτάται από τον όρο που θα επιλεχθεί από τη σχέση 4.10. Σημεία διακοπής (breakpoints) συμβαίνουν μόνο όταν το x συμπίπτει με τη δήλωση κάποιου άλλου παίκτη ή όταν οι δύο όροι μέσα στις αγκύλες της 4.10 που θεωρούνται συναρτήσεις του x, διασταυρώνονται. Έτσι ο αριθμός των διαστημάτων είναι πολυωνυμικός και το ολοκλήρωμα για κάθε διάστημα είναι μια έκφραση κλειστής μορφής, κι ο μηχανισμός που προκύπτει είναι υπολογίσιμος σε πολυωνυμικό χρόνο.

ΚΕΦΑΛΑΙΟ 5. ΑΝΑΘΕΣΗ ΔΙΕΡΓΑΣΙΩΝ ΣΕ ΜΗ ΣΥΣΧΕΤΙΖΟΜΕΝΕΣ ΜΗΧΑΝΕΣ ΜΕ ΠΕΡΙΟΡΙΣΜΟ ΩΣ ΠΡΟΣ ΤΗΝ ΠΡΟΣΒΑΣΗ

5.1. Πρόβλημα Ανάθεσης Διεργασιών σε Μη Συσχετιζόμενες Μηχανές

Εξετάζουμε το πρόβλημα της ανάθεσης διεργασιών για την περίπτωση των μη συσχετιζόμενων μηχανών. Το πρόβλημα ορίζεται ως εξής:

Επιθυμούμε την ανάθεση ενός συνόλου διεργασιών, N = {1,2,...,n}, με αδιαίρετο τρόπο, σε ένα σύνολο M={1,2,…,m} από **μη συσχετιζόμενες μηχανές** (unrelated machines). Αν η διεργασία j ανατεθεί στη μηχανή i, η πρόσθετη επιβάρυνση (φόρτος) που επιφέρει στη μηχανή αυτή η εκτέλεση της συγκεκριμένης διεργασίας είναι $p_{ij} \geq 0$ (χρόνος επεξεργασίας της διεργασίας j στη μηχανή i). Κάθε μηχανή, και μόνο αυτή, γνωρίζει το

χρόνο που χρειάζεται για να επεξεργαστεί μια συγκεκριμένη διεργασία. Ο συνολικός χρόνος επεξεργασίας (φόρτος) που απαιτεί μια μηχανή για να εκτελέσει όλες τις διεργασίες που τις έχουν ανατεθεί, ισούται με το άθροισμα των χρόνων επεξεργασίας όλων των διεργασιών που έχουν ανατεθεί στη συγκεκριμένη μηχανή.

Αναπαριστούμε μια ντετερμινιστική ανάθεση με το διάνυσμα $x = (x_{ij})_{i,j}$, όπου x_{ij} είναι 1 αν η διεργασία j ανατίθεται στη μηχανή i , κι έτσι έχουμε $x_{ij} \in \{0, 1\}$ για κάθε i, j, $\sum_i x_{ij} = 1$ για κάθε διεργασία j.

Στους πιθανοτικούς μηχανισμούς και στους μηχανισμούς που επιστρέφουν κλασματικές αναθέσεις, προσδιορίζουμε μια ανάθεση με το διάνυσμα $x = (x_{ij})_{i,j}$ με $\sum_i x_{ij} = 1$, αλλά εδώ $x_{ij} \in [0, 1]$ για κάθε i, j. Σε έναν πιθανοτικό μηχανισμό, με x_{ij} συμβολίζουμε την πιθανότητα η διεργασία j να ανατεθεί στη μηχανή i.

Αν A είναι το σύνολο όλων των δυνατών αναθέσεων των διεργασιών στις μηχανές, τότε για οποιαδήποτε ανάθεση $\alpha \in A$, κάθε μηχανή i επιβαρύνεται με ένα φορτίο (load), που ισούται με το συνολικό χρόνο εκτέλεσης όλων των διεργασιών που ανατίθενται σε αυτήν:

$$l_i = \sum_j x_{ij} p_{ij}$$

Κάθε μηχανή είναι ένας «εγωιστικά σκεπτόμενος» παίκτης, ο οποίος στοχεύει στην ελαχιστοποίηση του δικού του φόρτου εργασίας (συνολικού χρόνου επεξεργασίας). Ο μέγιστος φόρτος εργασίας (για μια συγκεκριμένη ανάθεση α) καλείται makespan της ανάθεσης: $\text{makespan}(\alpha) = \max_{i \in N}\{l_i(\alpha)\}$. Στόχος του προβλήματος είναι η εύρεση μιας ανάθεσης α^* που εξασφαλίζει τον ελάχιστο χρόνο περάτωσης: $\alpha^* \in \arg\min_{\alpha \in A}\{\text{makespan}(\alpha)\} = \arg\min_{\alpha \in A} \max_{i \in N}\{l_i(\alpha)\}$.

Πρέπει να σημειωθεί ότι, στις συσχετιζόμενες μηχανές ισχύει ότι $p_{ij} = p_j / s_i$ για κάθε i, j, όπου ο χρόνος εκτέλεσης της διεργασίας j, p_j, είναι δημοσίως γνωστός και η ταχύτητα s_i είναι η μόνη κρυφή παράμετρος της μηχανής i. Αυτό σημαίνει ότι το πεδίο ορισμού των τύπων των παικτών είναι ***μονοδιάστατο (single-dimensional)***, σε αντίθεση με το ***πολυδιάστατο πεδίο ορισμού (multidimensional domain)*** των μη συσχετιζόμενων μηχανών, όπου η κρυφή πληροφορία είναι ο χρόνος επεξεργασίας της διεργασίας j στη μηχανή i p_{ij}, δηλαδή ο τύπος που χαρακτηρίζει έναν παίκτη είναι ολόκληρο το διάνυσμα $(p_{ij})_j$, όπου $p_{ij} \in R_{\geq 0} \cup \{\infty\}$, που δηλώνει τους χρόνους επεξεργασίας της μηχανής i για κάθε διεργασία j με το ∞ να προσδιορίζει ότι η μηχανή i δε μπορεί να επεξεργαστεί την διεργασία j.

Ένας παίκτης, είναι δυνατό, να μη δηλώσει το διάνυσμα με τους πραγματικούς χρόνους επεξεργασίας, αν θεωρήσει ότι από κάτι τέτοιο θα επωφεληθεί μειώνοντας το κόστος του που ισούται με το συνολικό χρόνο που χρειάζεται για να επεξεργαστεί τις διεργασίες που του έχουν ανατεθεί (φόρτος εργασίας l_i). Στόχος είναι η κατασκευή φιλαληθών μηχανισμών για το παραπάνω πρόβλημα που επιτυγχάνουν μικρό λόγο προσέγγισης και είναι υπολογιστικά αποδοτικοί.

5.1.1. Πρόβλημα Ανάθεσης Διεργασιών με Περιορισμούς

Θεωρούμε τις ειδικές περιπτώσεις του προβλήματος ελαχιστοποίησης του makespan, όπου ο χρόνος επεξεργασίας μιας διεργασίας είναι είτε ¨Χαμηλός¨ (Low) είτε ¨Υψηλός¨ (High) σε κάθε μηχανή.

Συγκεκριμένα, $p_{ij} \in \{L_j, H_j\}$, για κάθε i, j, με $L_j \leq H_j$, όπου οι τιμές L_j, H_j είναι δημοσίως γνωστές ($L_j \equiv$ ¨ Low¨, $H_j \equiv$ ¨High¨). Αυτή είναι η περίπτωση των ***χρόνων επεξεργασίας δύο τιμών εξαρτώμενων από την διεργασία (job-dependent two-values case)***, που γενικεύει το κλασικό πρόβλημα των ¨περιορισμένων μηχανών¨ (restricted machines) όπου $p_{ij} \in \{L_j, \infty\}$, που έχει μελετηθεί αρκετά αλγοριθμικά.

Η δεύτερη ειδική περίπτωση είναι όταν έχουμε $L_j = L$, $H_j = H$ για όλες τις διεργασίες j, δηλαδή, $p_{ij} \in \{L, H\}$ για κάθε i, j, που ονομάζεται ***μοντέλο ανάθεσης δύο τιμών (¨two-values¨ scheduling model)***.

Και στις δύο παραπάνω περιπτώσεις έχουμε πολυδιάστατα πεδία ορισμού, αφού οι μηχανές είναι μη συσχετιζόμενες: μια διεργασία μπορεί να εκτελείται ως *χαμηλή* σε μια μηχανή $p_{ij} = L_j$ (στο εξής θα ονομάζεται *low)*, και ως *υψηλή* $p_{ij} = H_j$ (στο εξής θα ονομάζεται *high)*, σε μια άλλη. Παράλληλα κάποια άλλη διεργασία μπορεί να εκτελείται με διαφορετικό (αντίθετο) τρόπο. Επομένως, η κρυφή πληροφορία κάθε μηχανής είναι ένα διάνυσμα που προσδιορίζει ποιες διεργασίες είναι low σε αυτή και ποιες είναι high. Έτσι, διατηρείται η ιδιότητα που προσθέτει δυσκολία στη διαδικασία σχεδιασμού φιλαληθών μηχανισμών για μη συσχετιζόμενες μηχανές, και ευελπιστούμε ότι μελετώντας αυτές τις ειδικές περιπτώσεις θα αποκτήσουμε καλύτερη επίγνωση του γενικότερου προβλήματος.

5.1.2. Σχεδιασμός Μηχανισμών στις Μη Συσχετιζόμενες Μηχανές

Θεωρούμε μηχανισμούς άμεσης αποκάλυψης (x, P): κάθε μηχανή δηλώνει το (πιθανώς ψευδές) διάνυσμα $p = (p_{ij})_{i,j}$ με τους χρόνους επεξεργασίας της και στη συνέχεια ο μηχανισμός υπολογίζει αφενός μια ανάθεση διεργασιών $x(p)$, και αφετέρου τις αμοιβές $P = \{P_i(p)\}$ που θα δοθούν στις μηχανές, προκειμένου να αποζημιωθούν για το κόστος που επωμίζονται αναλαμβάνοντας την εκτέλεση των διεργασιών που τους ανατίθενται.

Στόχος του μηχανισμού είναι να υπολογίσει μια ανάθεση με σχεδόν βέλτιστο makespan σε σχέση με τους πραγματικούς χρόνους επεξεργασίας. ***Στόχος της μηχανής*** i είναι να μεγιστοποιήσει την ωφέλειά της $P_i - l_i$, που ισούται με την αποζημίωση που θα πάρει μείον το κόστος που θα προκύψει από τη συγκεκριμένη ανάθεση. Είναι δυνατό η μηχανή να δηλώσει ψευδείς χρόνους επεξεργασίας αν αυτό θα σήμαινε αύξηση της

ωφέλειάς της. Επομένως, ο μηχανισμός θα πρέπει, μέσω των αποζημιώσεων, να παρακινήσει τις μηχανές/παίκτες να αποκαλύψουν τους πραγματικούς χρόνους επεξεργασίας. Πράγμα που θα πετύχει χρησιμοποιώντας την φιλαλήθεια ως κυρίαρχη στρατηγική.

ΟΡΙΣΜΟΣ 5.1 (Φιλαλήθης Μηχανισμός): Ένας μηχανισμός ανάθεσης είναι φιλαλήθης αν, για κάθε μηχανή i, για κάθε διάνυσμα χρόνων επεξεργασίας των άλλων μηχανών, p_{-i}, κάθε διάνυσμα πραγματικών χρόνων επεξεργασίας p_i^1 και κάθε άλλο διάνυσμα p_i^2 της μηχανής i, έχουμε:

$$P_i(p_{-i}, p_i^1) - \sum_j p_{ij}{}^1 x_{ij}(p_{-i}, p_i^1) \geq P_i(p_{-i}, p_i^2) - \sum_j p_{ij}{}^1 x_{ij}(p_{-i}, p_i^2) \qquad \text{(σχέση 5.1)}$$

όπου $x_{ij}(p_{-i}, p_i^1)$, $x_{ij}(p_{-i}, p_i^2)$ οι αναθέσεις και $P_i(p_{-i}, p_i^1)$, $P_i(p_{-i}, p_i^2)$ οι αποζημιώσεις όταν οι άλλες μηχανές δηλώνουν p_{-i} και η μηχανή i δηλώνει p^1 και p^2 αντίστοιχα.

Στους κλασματικούς μηχανισμούς η φιλαλήθεια ορίζεται με τον ίδιο τρόπο, με τη διαφορά ότι στον ορισμό 5.1 τα $x_{ij}(p_{-i}, p_i^1)$ και $x_{ij}(p_{-i}, p_i^2)$ είναι οι κλασματικές αναθέσεις. Όσον αφορά, τους πιθανοτικούς μηχανισμούς, θεωρούμε ότι μια μηχανή μεγιστοποιεί την *αναμενόμενη ωφέλειά της (expected utility)* δηλώνοντας το διάνυσμα με τους πραγματικούς χρόνους επεξεργασίας. Η ανισότητα (5.1) καθορίζει την φιλαλήθεια *κατά αναμενόμενη τιμή (truthfulness-in-expectation)*, όταν τα $P_i(p_{-i}, p_i^1)$, $P_i(p_{-i}, p_i^2)$ συμβολίζουν τις αναμενόμενες αποζημιώσεις που καταβάλλονται στον παίκτη i, και τα $x_{ij}(p_{-i}, p_i^1)$, $x_{ij}(p_{-i}, p_i^2)$ είναι οι κλασματικές αναθέσεις που συμβολίζουν τις τυχαίες αναθέσεις του αλγόριθμου ($x_{ij}{}^k$: είναι η πιθανότητα να ανατεθεί η διεργασία j στη μηχανή i στην ανάθεση που θα προκύψει για τη δήλωση (p_{-i}, p_i^k)).

Η παραπάνω ανισότητα σημαίνει ότι αν η φιλαλήθεια είναι η κυρίαρχη στρατηγική των μηχανών, δεν υπάρχει καμία μηχανή που να βελτιώνει την ωφέλειά της δηλώνοντας ψευδή χρόνο επεξεργασίας, ανεξάρτητα με το τι έχουν δηλώσει οι άλλες μηχανές.

5.2. Κυκλική Μονοτονικότητα (Cycle Monotonicity)

Αρχικά, ο Rochet το 1987 παρατήρησε ότι η φιλαλήθεια για τα μονοδιάστατα πεδία ορισμού καθορίζεται από μια αλγοριθμική συνθήκη μονοτονικότητας, που ονομάζεται κυκλική μονοτονικότητα. Σε αυτήν την εργασία θα περιγράψουμε πως η συνθήκη της κυκλικής μονοτονικότητας μπορεί να χρησιμοποιηθεί αποδοτικά για να κατασκευάσουμε φιλαλήθεις μηχανισμούς σε πολυδιάστατα πεδία ορισμού.

Η συνθήκη της κυκλικής μονοτονικότητας είναι καλύτερο να οριστεί στο πρόβλημα της κοινωνικής επιλογής, επομένως, χρησιμοποιούμε το συμβολισμό της παραγράφου 2.4.1. Έχουμε ένα σύνολο A από εναλλακτικές καταστάσεις/επιλογές, υπάρχουν n παίκτες και κάθε παίκτης έχει μια κρυφή πληροφορία, τον τύπο (συνάρτηση αποτίμησης) v: $A \mapsto R$, όπου $v_i(\alpha)$ είναι η αποτίμηση του i για την εναλλακτική α. Υπενθυμίζουμε ότι στο πρόβλημα της ανάθεσης, το A αντιπροσωπεύει όλες τις δυνατές αναθέσεις των διεργασιών στις μηχανές, και το $v_i(\alpha)$ είναι η αρνητική τιμή του φορτίου της μηχανής i στην ανάθεση α. Έστω ότι V_i είναι το σύνολο όλων των δυνατών τύπων του παίκτη i. Έχουμε τον μηχανισμό (f, $\{P_i\}$) όπου $f : V_1 \times ... \times V_m \mapsto A$ είναι ο αλγόριθμος επιλογής της εναλλακτικής και $P_i : V_1 \times ... \times V_m \mapsto A$ είναι η τιμή με την οποία χρεώνεται ο παίκτης i (στο πρόβλημα της ανάθεσης ο μηχανισμός πληρώνει τους παίκτες).

Η σχέση 2.2 ορίζει την έννοια του φιλαλήθη μηχανισμού. Αν θέσουμε $\alpha = f(v_{-i},v_i)$ και $b = f(v_{-i},v_i')$ η ανισότητα γίνεται $v_i(\alpha) - P_i(v_{-i},v_i) \geq v_i(b) - P_i(v_{-i},v_i')$ και η ερώτηση είναι αν υπάρχει κάποιο σχήμα κοστολόγησης που να κάνει τον μηχανισμό φιλαλήθη.

Οι τιμές P_i μπορούν να εξαρτώνται μόνο από την κατάσταση που θα επιλεγεί και από τις δηλώσεις των άλλων παικτών, αυτό σημαίνει ότι μπορούμε να γράψουμε $P_i : V_{-i} \times A \mapsto R$. Επομένως, η φιλαλήθεια συνεπάγεται ότι για κάθε παίκτη i, κάθε $v_{-i} \in V_{-i}$ και κάθε $v_i, v_i' \in V_i$ με α = f(ν_{-i},ν_i) και b = f(ν_{-i},ν_i'), έχουμε $\nu_i(\alpha) - P_i(\nu_{-i}, \alpha) \geq \nu_i(b) - P_i(\nu_{-i}, b)$.

Για έναν παίκτη i, και για σταθερές δηλώσεις $\mathbf{v}_{-i}$ των άλλων παικτών το σύνολο A των εναλλακτικών επιλογών γίνεται A΄ = f (V_i, v_{-i}). Ψάχνουμε μια ανάθεση στις μεταβλητές $\{P_\alpha\}_{\alpha \epsilon A'}$ τέτοια ώστε $\nu_i(\alpha) - \nu_i(b) \geq P_\alpha - P_b$ για κάθε α, b ϵ A΄ και $v_i \in V_i$. Ορίζουμε την ποσότητα $\delta_{\alpha,b}$:= inf{ $\nu_i(\alpha) - \nu_i(b)$: $v_i \in V_i$, f($\mathbf{v}_{-i}$,ν_i) = α}. Τώρα μπορούμε να αναδιατυπώσουμε το πρόβλημα ανάθεσης τιμών: ψάχνουμε μια ανάθεση στις μεταβλητές $\{P_\alpha\}_{\alpha \epsilon A'}$ τέτοια ώστε

$$P_\alpha - P_b \leq \delta_{\alpha,b} \qquad \forall a, b \in A' \qquad \text{(σχέση 5.2)}$$

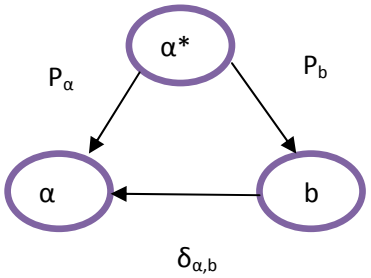
Το παραπάνω πρόβλημα λύνεται μελετώντας το γράφημα αναθέσεων (allocation graph) από τη Θεωρία Γραφημάτων (Graph theory).

ΟΡΙΣΜΟΣ 5.2: Το γράφημα ανάθεσης του αλγόριθμου f (συνάρτηση κοινωνικής επιλογής)είναι ένα κατευθυνόμενο σταθμισμένο γράφημα (directed weighted graph) G = (A, E) όπου E = A×A (κάθε κόμβος αντιστοιχεί σε μια συγκεκριμένη κατάσταση/ ανάθεση) και το βάρος μιας ακμής $b \to a$ (για κάθε $a, b \in A$) είναι ίσο με $\delta_{\alpha,b}$.

ΘΕΩΡΗΜΑ 5.3 [Κύκλοι Μη Αρνητικού Μήκους]: Υπάρχει μια εφικτή ανάθεση που να ικανοποιεί τη σχέση (5.2) αν και μόνο αν το γράφημα ανάθεσης δεν έχει κύκλους αρνητικού μήκους. Αν όλοι οι κύκλοι είναι μη αρνητικοί, μια εφικτή ανάθεση προκύπτει ως εξής: όρισε ένα τυχαίο κόμβο α* ϵ A και θέσε το P_α να είναι το μήκος της συντομότερης διαδρομής από το α* στο α[LaSw07].

Απόδειξη: Αρχικά θεωρούμε ότι υπάρχει μια εφικτή ανάθεση $\{P_\alpha\}_{\alpha \epsilon A}$ για την σχέση 5.2. Για οποιονδήποτε κύκλο $\alpha_1 \leftarrow \alpha_2 \ldots \leftarrow \alpha_k \leftarrow \alpha_1$, αν προσθέσουμε τις ανισότητες, $P_{\alpha_k} - P_{a_{k+1}} \leq \delta_{\alpha_k, a_{k+1}}$ για k = 1, …K, όπου $K+1 \equiv 1$, έχουμε ότι $\sum_{k=1}^{K} \delta_{\alpha_k, a_{k+1}} \geq 0$, που δείχνει ότι το μήκος του κύκλου είναι μη αρνητικό.

Στη συνέχεια θεωρούμε ότι το μήκος οποιουδήποτε κύκλου είναι μη αρνητικό. Ορίζουμε έναν τυχαίο κόμβο α* ϵ A θέτουμε το P_α να είναι το μήκος της συντομότερης διαδρομής από το α* στο α. Αφού δεν υπάρχουν αρνητικοί κύκλοι, η απόσταση της συντομότερης διαδρομής από το α* στο α είναι καλά ορισμένη και το P_α είναι πεπερασμένο για όλα τα α ϵ A. Συνεπάγεται ότι για όλα τα α, b ϵ A, έχουμε $P_\alpha - P_b \leq \delta_{\alpha,b}$, αφού η απόσταση της συντομότερης διαδρομής από το α* στο α είναι το πολύ $\delta_{\alpha,b}$ συν την απόσταση της συντομότερης διαδρομής από το α* στο b: $P_\alpha \leq \delta_{\alpha,b} + P_b$.



Το θεώρημα οδηγεί στον ακόλουθο ορισμό που είναι ένας άλλος τρόπος για να εκφραστεί η ιδιότητα ότι το γράφημα ανάθεσης δεν έχει αρνητικούς κύκλους.

ΟΡΙΣΜΟΣ 5.4 (Κυκλική Μονοτονικότητα): Μια συνάρτηση κοινωνικής επιλογής f ικανοποιεί την ιδιότητα της κυκλικής μονοτονικότητας αν για κάθε παίκτη i, κάθε $v_{-i} \in V_{-i}$, κάθε ακέραιο K, και κάθε $v_i^1, \ldots, v_i^k \in V_i$,

$\sum_{k=1}^{K} [v_i^k(a_k) - v_i^k(a_{k+1})] \geq 0$, όπου $\alpha_\kappa = f(\mathbf{v}_{-i}, v_i^k)$ για $1 \leq k \leq K$, και $a_{k+1} = a_1$

ΠΟΡΙΣΜΑ 5.5: Υπάρχουν αποζημιώσεις P τέτοιες ώστε ο μηχανισμός (f, P) να είναι φιλαλήθης αν και μόνο αν η f ικανοποιεί την κυκλική μονοτονικότητα.

Για το πρόβλημα της ανάθεσης διεργασιών σε μηχανές ορίζουμε τα i, p_{-i} και $p_i^1,...,p_i^k$ (χρόνοι επεξεργασίας). Έστω ότι $x(p_i^k, p_{-i}) = x^k$ για $1 \le k \le K$, και έστω $x^{k+1} = x^1$, $p^{k+1} = p^1$. Το x^k θα μπορούσε να ήταν μια $\{0,1\}$- ανάθεση ή μια κλασματική ανάθεση. Έχουμε $v_i^k(x^k) = -\sum_j x_{ij}^k p_{ij}^k$ όταν ο παίκτης δηλώνει τον πραγματικό χρόνο επεξεργασίας p_{ij}^k της διεργασίας j στη μηχανή i (προκύπτει η ανάθεση x^k) και $v_i^k(x^{k+1}) = -\sum_j x_{ij}^{k+1} p_{ij}^k$ όταν δηλώνει ψευδή χρόνο επεξεργασίας p_{ij}^{k+1} (προκύπτει η ανάθεση x^{k+1}). Επομένως η κυκλική μονοτονικότητα ερμηνεύεται ως εξής $\sum_{k=1}^{K}[-\sum_j x_{ij}^k p_{ij}^k + \sum_j x_{ij}^{k+1} p_{ij}^k] \ge 0$. Αναδιατάσσοντας,

$$\sum_{k=1}^{K} \sum_j x_{ij}^{k+1}(p_{ij}^k - p_{ij}^{k+1}) \ge 0 \qquad \text{(σχέση 5.3)}$$

Έτσι, το πρόβλημα του σχεδιασμού μηχανισμών ανάγεται σε ένα αλγοριθμικό πρόβλημα. Θα προσπαθήσουμε να σχεδιάσουμε έναν προσεγγιστικό αλγόριθμο για το πρόβλημα ελαχιστοποίησης του makespan που να ικανοποιεί την σχέση 5.3.

5.3. Γενική Τεχνική Κατασκευής Πιθανοτικών Μηχανισμών

Στην περίπτωση των πολυδιάστατων πεδίων ορισμού, υπάρχει φιλαλήθης μηχανισμός που να υλοποιεί το πρόβλημα ελαχιστοποίησης του makespan βέλτιστα, επομένως θα χρησιμοποιηθούν προσεγγιστικοί αλγόριθμοι και θα περιγραφθεί μια γενική τεχνική που μετατρέπει έναν τέτοιον c- προσεγγιστικό αλγόριθμο σε 3c- προσεγγιστικό και φιλαλήθη κατά αναμενόμενη τιμή μηχανισμό (3c-approximation, truthful-in-expectation mechanism).

Η απόδειξη είναι απλή και βασίζεται σε δύο ιδέες: Πρώτον, αποδεικνύεται η φιλαλήθεια των παικτών χρησιμοποιώντας την ιδιότητα της κυκλικής μονοτονικότητας. Δεύτερον, η κατασκευή του πιθανοτικού μηχανισμού περιλαμβάνει α) αρχικά τη μετατροπή του πιθανοτικού προσεγγιστικού μηχανισμού σε έναν κλασματικό φιλαλήθη μηχανισμό που επιστρέφει κλασματικές αναθέσεις και β) στη συνέχεια μια διαδικασία στρογγυλοποίησης (rounding procedure) προκειμένου να εκφραστεί η κλασματική ανάθεση σαν ένας κυρτός συνδυασμός ακέραιων αναθέσεων.

ΛΗΜΜΑ 5.6: Έστω ότι έχουμε έναν κλασματικό φιλαλήθη μηχανισμό M = (x, P). Έστω R ένας πιθανοτικός αλγόριθμος στρογγυλοποίησης, που δεδομένης μιας κλασματικής ανάθεσης x, εξάγει μια τυχαία ανάθεση R(x) (τυχαίες μεταβλητές {0, 1}) τέτοια ώστε $E[R(x)_{ij}] = x_{ij}$ (κλασματικές αναθέσεις) για όλα τα i, j. Όρισε το X(p) να είναι η τυχαία ανάθεση R(x(p)). Υπάρχουν αποζημιώσεις P΄ τέτοιες ώστε ο μηχανισμός M΄ = (X, P΄) να είναι φιλαλήθης κατά αναμενόμενη τιμή. Επιπλέον, αν ο M έχει την ιδιότητα της οικειοθελούς συμμετοχής τότε και ο M' έχει την ιδιότητα της οικειοθελούς συμμετοχής για κάθε ρίψη του νομίσματος.

Απόδειξη: Γνωρίζουμε ότι X(p) είναι η τυχαία ανάθεση R(x(p)) που ικανοποιεί την $E[X(p)_{ij}] = x(p)_{ij}$ για όλα τα i, j. $P_i(p)$ είναι οι αποζημιώσεις που δίνονται στον παίκτη i όταν η είσοδος είναι p (χρόνος επεξεργασίας). Πρέπει να ορίσουμε τις πιθανώς τυχαίες αποζημιώσεις $P_i'(p)$ σε κάθε παίκτη i έτσι ώστε M' = (X, P') να είναι ένας φιλαλήθης κατά αναμενόμενη τιμή μηχανισμός.

Για μια είσοδο p, για τη μηχανή i και την ανάθεση x, ορίζουμε το $p_i(x) = \sum_j x_{ij} p_{ij}$.

Παρατηρούμε ότι για οποιαδήποτε είσοδο p^1 και p^2, έχουμε $E[p^1(X(p^2))] = p^1(E[X(p^2)] = p^1(x(p^2))$, όπου η πρώτη ισότητα προκύπτει από τη γραμμικότητα του p_i^1, και η δεύτερη προκύπτει από το ότι $E[X(p^2)_{ij}] = x(p^2)_{ij}$ για κάθε i, j. Έτσι, αν θέσουμε τις αποζημιώσεις $P_i'(\sqcup)$ έτσι ώστε $E[P_i'(p)] = P_i(p)$ για κάθε είσοδο p, η αναμενόμενη ωφέλεια ενός παίκτη i, όταν η πραγματική του τιμή είναι p^1 και αυτή που

δηλώνει είναι p^2, θα ισούται με την ωφέλεια που έχει στο μηχανισμό M όταν η πραγματική του τιμή είναι p^1 και αυτή που δηλώνει είναι p^2. Συνεπάγεται ότι αφού ο μηχανισμός M είναι φιλαλήθης, ο M' θα είναι φιλαλήθης κατά αναμενόμενη τιμή.

Η συνθήκη $E[P_i'(p)] = P_i(p)$ δίνει ευελιξία στην κατασκευή αποζημιώσεων, για παράδειγμα θα μπορούσαμε απλά να θέσουμε $P_i'(p) = P_i(p)$. Το σχήμα κοστολόγησης που θα πρέπει να μεταφέρει την ιδιότητα *της οικειοθελούς συμμετοχής* από τον μηχανισμό M στον M΄. Πράγμα που επιτυγχάνεται ως εξής: Έστω $\{y^{(1)}(p), \ldots, y^{(k)}(p)\}$ όλες οι πιθανές τιμές (ακέραιες αναθέσεις) της τυχαίας μεταβλητής X(p). Θέτουμε το $P'_i(p)$ ίσο με την τυχαία μεταβλητή:

$$P'_i(p) = \begin{cases} p_i(y^{(l)}(p))\dfrac{P_i(p)}{p_i(x(p))} & \text{αν } p_i(x(p))>0 \text{ και } X(p) = y^{(l)}(p) \\ P_i(p) & \text{αν } p_i(x(p)) = 0 \end{cases}$$

Ισχύει ότι $E[P_i'(p)] = P_i(p)$. Αν ο M ικανοποιεί την ιδιότητα της *οικειοθελούς συμμετοχής*, έχουμε ότι $P_i(p) \geq p_i(x(p))$ για κάθε είσοδο p. Αυτό σημαίνει ότι θα ισχύει πάντα $P_i'(p) \geq p_i(X(p))$ και άρα ο μηχανισμός M΄ ικανοποιεί την ιδιότητα *της οικειοθελούς συμμετοχής*.

ΛΗΜΜΑ 5.7 [Διαδικασία Στρογγυλοποίησης]: Δεδομένης μιας κλασματικής ανάθεσης x κι ενός διανύσματος χρόνων επεξεργασίας p, υπάρχει μια πιθανοτική διαδικασία στρογγυλοποίησης (randomized rounding procedure) που εξάγει μια τυχαία ανάθεση X τέτοια ώστε,

1. Για κάθε i, j, $E[X_{ij}] = x_{ij}$
2. Για κάθε i, $\sum_j X_{ij} p_{ij} < \sum_j x_{ij} p_{ij} + \max_{\{j:x_{ij}>0\}} p_{ij}$ με πιθανότητα 1

Η ιδιότητα 1 θα χρησιμοποιηθεί για να αποκτήσουμε φιλαλήθεις παίκτες κατά αναμενόμενη τιμή και η ιδιότητα 2 θα βοηθήσει στην απόδειξη της προσεγγιστικότητας του αλγόριθμου.

Απόδειξη: Έστω $n_i = \left\lceil \sum_j x_{ij} \right\rceil$. Δεδομένης της κλασματικής ανάθεσης x, θα δημιουργήσουμε ένα διμερές γράφημα G = (V,W,E). Το W είναι ένα σύνολο από κόμβους-διεργασίες, ένας κόμβος w_j αντιστοιχεί σε κάθε διεργασία j, και V είναι ένα σύνολο από κόμβους-μηχανές, για κάθε μηχανή i, δημιουργούμε n_i κόμβους $v_{i,c}$, όπου c = 1, . . . , n_i. Οι ακμές δημιουργούνται ως εξής: για κάθε μηχανή i, διατάσσουμε τις διεργασίες με $x_{ij} > 0$ κατά φθίνουσα σειρά σε σχέση με τους χρόνους επεξεργασίας τους p_{ij}. Από το ¨αντίγραφο¨ $v_{i,1}$ δημιουργούμε μια ακμή ($v_{i,1}$, w_j) για κάθε διεργασία j μέχρι το συνολικό βάρος x_{ij} αυτών των διεργασιών να γίνει τουλάχιστον 1. Στη συνέχεια παίρνουμε το επόμενο αντίγραφο $v_{i,2}$, και από τις διεργασίες που απέμειναν δημιουργούμε μια ακμή ($v_{i,2}$,w_j) για κάθε διεργασία μέχρι το συνολικό βάρος x_{ij} αυτών των διεργασιών να ξεπεράσει το 1, και συνεχίζουμε με αυτόν τον τρόπο.

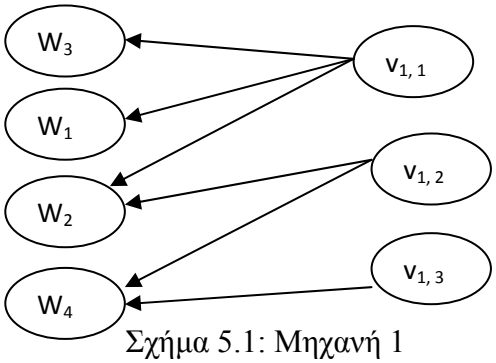
Αν για κάποιο αντίγραφο $v_{i,c}$, το συνολικό βάρος x_{ij} των διεργασιών με ακμές στο $v_{i,c}$ υπερβαίνει το 1, τότε η τελευταία στη διάταξη διεργασία k που έχει μια ακμή ($v_{i,c}$,w_k) και συμβαίνει κάτι τέτοιο, θα είναι η πρώτη διεργασία που θα ανατεθεί στο $v_{i,c+1}$ με βάρος ίσο με το αρχικό βάρος x_{ik} της διεργασίας μείον το ποσοστό της διεργασίας που ανατέθηκε στο αντίγραφο $v_{i,c}$. Έτσι, η διεργασία k διαμερίζεται μεταξύ δύο αντιγράφων $v_{i,c}$ και $v_{i,c+1}$.

Έστω ότι έχουμε το παρακάτω παράδειγμα:

Κλασματικές αναθέσεις (x_{ij})	**Διεργασίες j**				$\sum_j x_{ij}$	**n_i**
Μηχανές i	**1**	**2**	**3**	**4**		
1	0,2	0,6	0,8	0,5	2,1	3
2	0,5	0,2	0,2	0,4	2	2
3	0,3	0,2	0	0,1	1	1

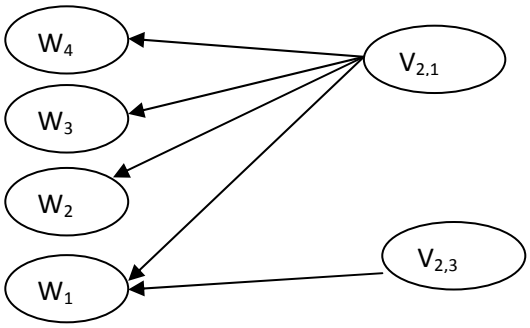
Χρόνοι Επεξεργασίας (p_{ij})	**Διεργασίες j**			
Μηχανές i	**1**	**2**	**3**	**4**
1	2	1	3	1
2	1	2	3	4
3	4	3	2	1

Για τη **Μηχανή 1** κατατάσσουμε τις διεργασίες κατά φθίνουσα σειρά σε σχέση με τα p_{ij}. Σύμφωνα με τον παραπάνω πίνακα έχουμε W_3 W_1 W_2 W_4. Δημιουργούμε n_i =3 κόμβους $v_{1,c}$

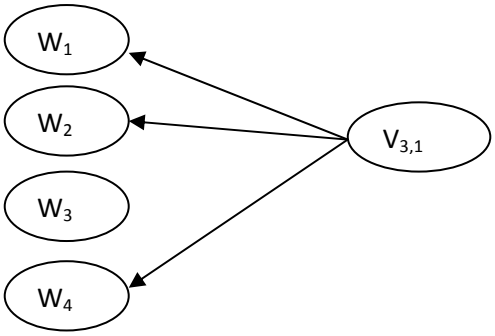


Σχήμα 5.1: Μηχανή 1

Παρομοίως, φτιάχνουμε το γράφημα για τη Μηχανή 2 και 3 αντίστοιχα.



Σχήμα 5.2: Μηχανή 2



Σχήμα 5.3: Μηχανή 3

Συνοψίζουμε κάποιες ιδιότητες του διμερούς γραφήματος:

1. Το γράφημα G περιέχει μια ακμή $(v_{i,c},w_j)$ για κάποιο αντίγραφο $v_{i,c}$ αν και μόνο αν $x_{ij} > 0$. Για κάθε i, j με $x_{ij} > 0$, οι ακμές που προσπίπτουν στον κόμβο w_j αποτελούν είτε (i) μια ακμή της μορφής $(v_{i,c},w_j)$, ή (ii) δύο ακμές της μορφής $(v_{i,c},w_j)$, $(v_{i,c+1},w_j)$.

2. Για κάθε μηχανή i και ακμές $(v_{i,c},w_j)$, $(v_{i,c'},w_k)$, όπου $c < c'$, έχουμε $p_{ij} \geq p_{ik}$, επειδή οι διεργασίες ταξινομούνται κατά φθίνουσα σειρά.

3. Οποιοδήποτε ταίριασμα στο G, που ταιριάζει (matches)[8] όλους τους κόμβους-διεργασίες, αντιστοιχίζεται σε μια ακέραια ανάθεση, όπου μια διεργασία j ανατίθεται σε μια μηχανή i αν και μόνο αν, στο ταίριασμα υπάρχει μια ακμή της μορφής $(v_{i,c}, w_j)$. Επιπλέον, αυτή η ανάθεση ικανοποιεί την ιδιότητα 2, που σημαίνει ότι το φορτίο σε κάθε μηχανή i είναι το πολύ

$$\sum_j X_{ij} p_{ij} < \sum_j x_{ij} p_{ij} + \max_{\{j:x_{ij}>0\}} p_{ij}$$

[8] Μια εργασία εκτελείται σε μια μηχανή

4. Η ανάθεση x μετατρέπεται σε ένα κλασματικό ταίριασμα στο γράφημα που ταιριάζει εξολοκλήρου όλους τους κόμβους-διεργασίες και όλους τους κόμβους $v_{i,c}$, c = 1, . . . , n_i−1. Το συνολικό βάρος που ανατίθεται στις ακμές της μορφής $(v_{i,c},w_j)$ από αυτό το κλασματικό ταίριασμα είναι ακριβώς x_{ij}.

Η ιδιότητα 3 είναι επακόλουθο της ιδιότητας 2: Έστω M^F το κλασματικό ταίριασμα που προέκυψε από την ανάθεση x, η οποία αναθέτει βάρη $M^F_{v_{i,c},w_j}$ στις ακμές $(v_{i,c}, w_j)$. Αν y είναι η ανάθεση που αντιστοιχεί σε ένα ακέραιο ταίριασμα M, που ταιριάζει όλους τους κόμβους- διεργασίες, τότε το φορτίο στη μηχανή i είναι το πολύ

$$\max_{\{j:x_{ij}>0\}} p_{ij} + \sum_{c=2}^{n_i} \sum_{j:(v_{i,c},w_j)\in M} p_{ij} \le \max_{\{j:x_{ij}>0\}} p_{ij} + \sum_{c=2}^{n_i} \sum_{j:(v_{i,c-1},w_j)\in E} M^F_{v_{i,c},w_j} p_{ij}$$

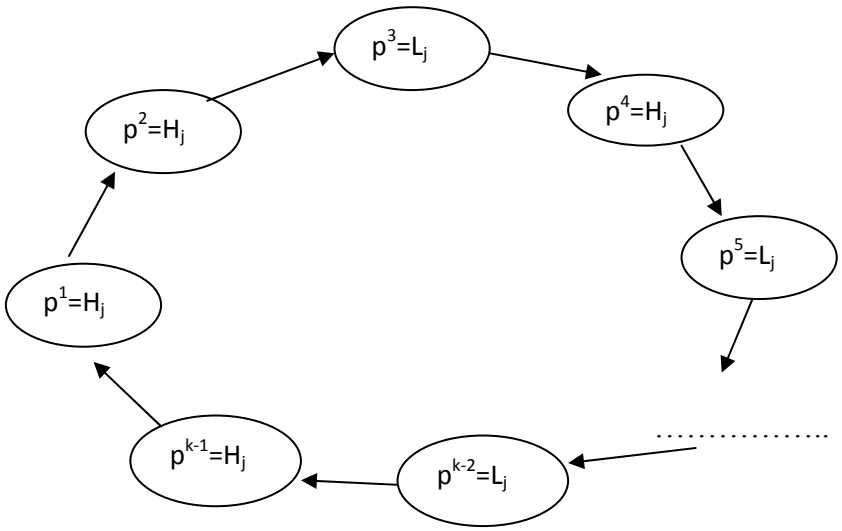
$$\le \max_{\{j:x_{ij}>0\}} p_{ij} + \sum_{j:x_{ij}>0} x_{ij} p_{ij}$$

Θα αποδείξουμε ότι οποιοσδήποτε αλγόριθμος επιστρέφει μια κλασματική ανάθεση που έχει συγκεκριμένες ιδιότητες ικανοποιεί την ιδιότητα της κυκλικής μονοτονικότητας.

ΛΗΜΜΑ 5.8: Έστω A ένας αλγόριθμος που για κάθε είσοδο p, εξάγει μια κλασματική ανάθεση x τέτοια ώστε, αν $p_{ij} = H_j$ τότε $x_{ij} \le 1/m$, και αν $p_{ij} = L_j$ τότε $x_{ij} \ge 1/m$. Τότε ο αλγόριθμος A ικανοποιεί την κυκλική μονοτονικότητα.

Απόδειξη: Θεωρούμε έναν παίκτη i, το διάνυσμα των χρόνων επεξεργασίας των άλλων παικτών p_{-i}, και θέλουμε να αποδείξουμε ότι ισχύει η (4.3) $\sum_{k=1}^{K} \sum_j x_{ij}^{k+1}(p_{ij}^{k} - p_{ij}^{k+1}) \ge 0$, για κάθε p^1_i , . . . , p^K_i , όπου k = K + 1 = 1. Αρκεί να δείξουμε ότι $\sum_{k=1}^{K} x_{ij}^{k+1}(p_{ij}^{k} - p_{ij}^{k+1}) \ge 0$.

Αν $p_{ij}^{\ k} = L$ για όλα τα k ή $p_{ij}^{\ k} = H$ για όλα τα k η παραπάνω σχέση ισχύει ως ισότητα. Αλλιώς, για τον παίκτη i, έστω ότι έχουμε τον παρακάτω κύκλο αναθέσεων.



Σχήμα 5.4: Κύκλος Αναθέσεων

Δε μας ενδιαφέρουν οι μεταβάσεις κατά τις οποίες δε μεταβάλλεται ο χρόνος επεξεργασίας που δηλώνει ο παίκτης, καθώς οι όροι που δημιουργούνται συνεισφέρουν μηδενικά στο άθροισμα: $x_{ij}^{\ k+1}(H_j - H_j) = 0$ και $x_{ij}^{\ k+1}(L_j - L_j) = 0$ (βλ. μεταβάσεις $\mathrm{p}^1 \to p^2, \mathrm{p}^{k-1} \to p^1$ κ.ο.κ. στο σχήμα 5.4). Επομένως, μας ενδιαφέρει το παρακάτω άθροισμα:

$$\sum_{L\to H} x^{k+1}{}_{ij}(L_j - H_j) + \sum_{H\to L} x^{k+1}{}_{ij}(H_j - L_j) = \left(\sum_{H\to L} x^{k+1}{}_{ij} - \sum_{L\to H} x^{k+1}{}_{ij}\right)(H_j - L_j) \geq 0$$

Σε έναν κύκλο ο αριθμός των μεταβάσεων $H \to L$ ισούται με τον αριθμό των μεταβάσεων $L \to H$, και δεδομένου ότι αν $p_{ij} = H_j$ τότε $x_{ij} \leq 1/m$, και αν $p_{ij} = L_j$ τότε $x_{ij} \geq 1/m$, το παραπάνω άθροισμα είναι μη αρνητικό αφού $\sum_{H\to L} x^{k+1}{}_{ij} \geq \sum_{L\to H} x^{k+1}{}_{ij}$.

Παρακάτω περιγράφουμε πως χρησιμοποιούμε έναν c-προσεγγιστικό αλγόριθμο, που επιστρέφει μια ακέραια ανάθεση, για να αποκτήσουμε έναν αλγόριθμο που να ικανοποιεί την ιδιότητα του λήμματος 5.8. Παίρνουμε κάθε διεργασία j που ανατίθεται σε μια high μηχανή και αναθέτουμε 1/m μέρος αυτής της διεργασίας σε όλες τις μηχανές. Για κάθε διεργασία j που ανατίθεται σε low[9] μηχανή, έστω την i, αναθέτουμε 1/m ποσοστό αυτής σε οποιαδήποτε άλλη μηχανή εκτελεί τη συγκεκριμένη διεργασία σαν low, και αναθέτουμε το κλάσμα που απομένει (που είναι τουλάχιστον 1/m) στην i. Η ανάθεση που προκύπτει ικανοποιεί τις επιθυμητές ιδιότητες. Επίσης, παρατηρούμε ότι το φορτίο σε κάθε μηχανή έχει αυξηθεί το πολύ κατά 1/m·(φορτίο στις άλλες μηχανές) ≤ makespan, και επομένως το makespan έχει το πολύ διπλασιαστεί.

Τέτοιου είδους ανάθεση μπορεί να γίνει και στην περίπτωση που η αρχική ανάθεση είναι κλασματική. Από τη στιγμή που μια διεργασία j μπορεί να ανατεθεί σε πολλές μηχανές, παίρνουμε κάθε μηχανή i για την οποία ισχύει ότι $x_{ij} > 0$, και διανέμουμε αυτό το κλάσμα ως εξής: αν η i είναι high μηχανή για την j, αναθέτουμε ένα κλάσμα x_{ij}/m της j σε όλες τις μηχανές, αν η μηχανή i είναι low για την διεργασία j, αναθέτουμε ένα κλάσμα x_{ij}/m της j σε όλες τις μηχανές που εκτελούν την διεργασία σαν low και το υπόλοιπο κλάσμα της διεργασίας $x_{ij}(1-\frac{\left|\{i' \neq i : p_{i'j} = L_j\}\right|}{m}$ το αναθέτουμε στην μηχανή i.

ΑΛΓΟΡΙΘΜΟΣ 1 [LaSw07]**:** Έστω A ένας οποιοσδήποτε αλγόριθμος που για οποιαδήποτε είσοδο p εξάγει μια πιθανώς κλασματική ανάθεση x τέτοια ώστε $x_{ij} > 0$ σημαίνει ότι pij ≤ makespan(x). Οποιοσδήποτε αλγόριθμος επιστρέφει μια ακέραια ανάθεση έχει αυτές τις ιδιότητες. Ο Αλγόριθμος 1 επιστρέφει τις ακόλουθες αναθέσεις x^F. Για κάθε i, j,

1. Αν $p_{ij} = H_j$, θέσε $x_{ij}^F = \sum_{i':p_{i'j}=H_j} x_{i'j} / m$

[9] Θα λέμε ότι μια εργασία j ανατίθεται σε μια low μηχανή όταν $p_{ij} = L$

2. Αν $p_{ij} = L_j$, θέσε $x_{ij}^F = x_{ij} + \sum_{i' \neq i : p_{i'j} = L_j} (x_{i'j} - x_{ij}) / m + \sum_{i' : p_{i'j} = H_j} x_{i'j} / m$

ΘΕΩΡΗΜΑ 5.9 [Ο Αλγόριθμος 1 είναι 2c- Προσεγγιστικός]: Έστω ότι ο αλγόριθμος Α ικανοποιεί τις συνθήκες του Αλγορίθμου 1 κι επιστρέφει ένα makespan το πολύ ίσο με c·OPT(p)[10] για κάθε p. Τότε ο Αλγόριθμος 1 είναι ένας 2c- προσεγγιστικός, κυκλικά μονότονος και κλασματικός αλγόριθμος. Επιπλέον, αν $x_{ij}^F > 0$ κατά την είσοδο p, τότε $p_{ij} \leq c \cdot OPT(p)$.

Απόδειξη: Αρχικά, παρατηρούμε ότι η x^F είναι μια **έγκυρη ανάθεση**. Για κάθε διεργασία j έχουμε

$$\sum_i x_{ij}^F = \sum_{i:p_{ij}=H_j} x_{ij}^F + \sum_{i:p_{ij}=L_j} x_{ij}^F =$$

$$\sum_{i:p_{ij}=H_j} \sum_{k \neq i : p_{kj} = H_j} x_{kj} / m + \sum_{i:p_{ij}=L_j} x_{ij} + \sum_{i:p_{ij}=L_j} \sum_{k \neq i : p_{kj} = L_j} (x_{kj} - x_{ij}) / m + \sum_{i:p_{ij}=L_j} \sum_{k \neq i : p_{kj} = H_j} x_{kj} / m$$

(α)

Έχουμε ότι

$$\sum_{i:p_{ij}=H_j} \sum_{k \neq i : p_{kj} = H_j} x_{kj} / m + \sum_{i:p_{ij}=L_j} \sum_{k \neq i : p_{kj} = H_j} x_{kj} / m = \sum_i \sum_{i:p_{ij}=H_j} x_{ij} / m = \mathrm{m} \sum_{i:p_{ij}=H_j} x_{ij} / m = \sum_{i:p_{ij}=H_j} x_{ij}$$

(β)

$$\sum_{i:p_{ij}=H_j} x_{ij} + \sum_{i:p_{ij}=L_j} x_{ij} = \sum_i x_{ij}$$

(γ)

Από τις (α), (β) και (γ) έχουμε

[10] Το OPT(p) απεικονίζει το βέλτιστο (optimal) makespan για το στιγμιότυπο p

$$\sum_i x_{ij} + \sum_{i:p_{ij}=L_j} \sum_{k\neq i:p_{kj}=L_j} x_{kj}/m - \sum_{i:p_{ij}=L_j} \sum_{k\neq i:p_{kj}=L_j} x_{ij}/m = \sum_i x_{ij} = 1$$

<u>Ο Αλγόριθμος 1 ικανοποιεί τις συνθήκες κυκλικής μονοτονικότητας:</u>

Επίσης, έχουμε ότι αν $\mathbf{p_{ij} = H_j}$, τότε $x_{ij}^{F} = \sum_{k\neq i:p_{kj}=H_j} x_{kj}/m \leq 1/m$.

Αν $\mathbf{p_{ij} = L_j}$, τότε $x_{ij}^{F} = x_{ij}(1-l/m) + \sum_{k\neq i} x_{kj}/m$,

όπου $l = \left|\{k \neq i : p_{kj} = L_j\}\right| \leq$ m-1, δηλαδή ο αριθμός των low μηχανών χωρίς την i.

Έτσι,

$$x_{ij}^{F} \geq \sum_{k\neq i} x_{kj}/m = 1/m$$

Το συνολικό φορτίο οποιασδήποτε μηχανής i υπό την ανάθεση x^F είναι

$$\sum_{j:p_{ij}=H_j} \sum_{k\neq i:p_{kj}=H_j} H_j \cdot x_{kj}/m + \sum_{j:p_{ij}=L_j} L_j \cdot (x_{ij} + \sum_{k\neq i:p_{kj}=L_j} x_{kj}/m - \sum_{k\neq i:p_{kj}=L_j} x_{ij}/m + \sum_{k\neq i:p_{kj}=H_j} x_{kj}/m) =$$

$$\sum_{j:p_{ij}=H_j} \sum_{k\neq i:p_{kj}=H_j} H_j \cdot x_{kj}/m + \sum_{j:p_{ij}=L_j} L_j \cdot (x_{ij} + \sum_{k\neq i} x_{kj}/m - \sum_{k\neq i:p_{kj}=L_j} x_{ij}/m)$$

Επομένως, το συνολικό φορτίο οποιασδήποτε μηχανής i υπό την ανάθεση x^F είναι το πολύ

$$\sum_{j:p_{ij}=H_j} \sum_{k\neq i:p_{kj}=H_j} H_j \cdot x_{kj}/m + \sum_{j:p_{ij}=L_j} L_j \cdot (x_{ij} + \sum_{k\neq i} x_{kj}/m) =$$

$$[\sum_{j:p_{ij}=H_j}\sum_{k\neq i:p_{kj}=H_j} H_j \cdot x_{kj} / m + \sum_{j:p_{ij}=L_j} L_j \cdot x_{ij}\ ] + \sum_{j:p_{ij}=L_j} L_j \sum_{k\neq i} x_{kj} / m \leq$$

$$[\sum_{j:p_{ij}=H_j} m \cdot H_j \cdot x_{kj} / m + \sum_{j:p_{ij}=L_j} L_j \cdot x_{ij}\ ] + \sum_{k\neq i}\sum_{j:p_{ij}=L_j} L_j \cdot x_{kj} / m \leq$$

$$\sum_j p_{ij}x_{ij} + \sum_{k\neq i}\sum_j p_{kj}x_{kj} / m$$

<u>Ο Αλγόριθμος 1 είναι 2c- προσεγγιστικός αφού</u>

$$\sum_j p_{ij}x_{ij} + \sum_{k\neq i}\sum_j p_{kj}x_{kj} / m \leq 2c \cdot \mathrm{OPT(p)}$$

Αν $x_{ij}^{F} > 0$ κατά την είσοδο p, τότε pij ≤ c · OPT(p)

Έστω ότι $x_{ij}^{F} > 0$ για κάποια i και j:

- Αν $p_{ij} = L_j$, τότε είναι ξεκάθαρο ότι $p_{ij} \leq \mathrm{OPT(p)}$. Το OPT(p) είναι τουλάχιστον L.
- Αν $p_{ij} = H_j$, τότε για κάποιο i με $p_{ij} = H_j$ έχουμε $x_{ij} > 0$, και επομένως σύμφωνα με την υπόθεση *pij ≤ makespan(x)* του Αλγορίθμου 1, προκύπτει ότι *pij ≤ c · OPT(p)*. Δηλαδή, εξασφαλίζεται ότι μια διεργασία ανατίθεται σε μια μηχανή, αν ο συνολικός χρόνος επεξεργασίας της μηχανής που θα προκύψει μετά από την εκτέλεση και της συγκεκριμένης διεργασίας j είναι μικρότερος από το makespan(x).

Το θεώρημα 5.9 αν συνδυαστεί με τα λήμματα 5.6 και 5.7, δίνει έναν 3c-προσεγγιστικό και φιλαλήθη κατά αναμενόμενη τιμή μηχανισμό.

Συνοψίζοντας, ο μηχανισμός M′ = (X, P′) στο σύνολό του λειτουργεί ως εξής: δεδομένου ενός αλγόριθμου A, που ικανοποιεί τις συνθήκες του Αλγόριθμου 1, και μιας εισόδου p, ο μηχανισμός εξάγει την ακόλουθη τυχαία ανάθεση και τις ακόλουθες τιμές.

Ανάθεση X(p). Από την ανάθεση x=A(p), υπολογίζεται η ανάθεση x^F από τον Αλγόριθμο 1. Η τυχαία ανάθεση X(p) είναι η ανάθεση που προκύπτει χρησιμοποιώντας το λήμμα 4.7 για να στρογγυλοποιηθεί η x^F και να μεταβούμε σε ακέραιες αναθέσεις.

Αποζημιώσεις $\{P_i'(p)\}$. Έστω ότι υπάρχει αλγόριθμος Α΄΄ τέτοιος ώστε x^F = Α΄΄(p). Αρχικά, υπολογίζεται το σχήμα κοστολόγησης P έτσι ώστε ο μηχανισμός Μ΄΄= (Α΄΄, P) να είναι κλασματικός και φιλαλήθης: οι αποζημιώσεις $P_i(p)$ που δίνονται στον παίκτη i υπολογίζονται σύμφωνα με το Θεώρημα 4.3. Στη συνέχεια, χρησιμοποιείται το λήμμα 4.6 στον μηχανισμό Μ΄΄ για να προκύψουν οι τυχαίες αποζημιώσεις $P_i'(p)$.

ΘΕΩΡΗΜΑ 5.10: Η διαδικασία που περιλαμβάνει ο Αλγόριθμος 1 μετατρέπει οποιονδήποτε c-προσεγγιστικό και κλασματικό αλγόριθμο σε έναν 3c- προσεγγιστικό και φιλαλήθη κατά αναμενόμενη τιμή μηχανισμό.

Είναι γνωστό ότι το μικρότερο makespan T*, για το οποίο υπάρχει μια εφικτή κλασματική ανάθεση x που έχει την ιδιότητα $x_{ij} > 0 \Rightarrow p_{ij} \leq T^*$, μπορεί να υπολογιστεί σε πολυωνυμικό χρόνο. Παρατηρούμε ότι $T^* \leq OPT(p)$. Αν ο βασικός αλγόριθμος Α που θα χρησιμοποιήσουμε είναι βέλτιστος, αποκτούμε τελικά έναν 3-προσεγγιστικό μηχανισμό.

ΠΟΡΙΣΜΑ 5.11: Υπάρχει ένας μηχανισμός που εξασφαλίζει φιλαλήθεια κατά αναμενόμενη τιμή με λόγο προσέγγισης 3 για το Lj-Hj πρόβλημα ανάθεσης διεργασιών.

Παρατηρούμε ότι ακόμα και σε αυτές τις φαινομενικά απλές περιπτώσεις δεν υπάρχουν φιλαλήθεις μηχανισμοί που να επιστρέφουν μια βέλτιστη ανάθεση (σε αντίθεση με την περίπτωση των συσχετιζόμενων μηχανών).

5.4. Ντετερμινιστικός Μηχανισμός για την Περίπτωση Διεργασιών ¨Δύο-Τιμών¨

Θα παρουσιάσουμε έναν 2-προσεγγιστικό φιλαλήθη μηχανισμό για την περίπτωση όπου p_{ij} ϵ {L, H} για όλα τα i, j. Ο στόχος είναι να κατασκευαστεί ένας προσεγγιστικός αλγόριθμος που να ικανοποιεί την ιδιότητα της κυκλικής μονοτονικότητας. Το τεχνικό εργαλείο που θα χρησιμοποιηθεί είναι οι ***ροές δικτύων (network flows)***[Vohr07]**.**

Η ροή σε ένα δίκτυο προσδιορίζεται από ένα κατευθυνόμενο γράφημα G = (V, E) με ακμές e ϵ E που έχουν χωρητικότητα (capacity) $c_e \geq 0$, έναν κόμβο έναρξης (source) s ϵ V και έναν κόμβο προορισμού (sink) t ϵ V. Μια ροή είναι μια συνάρτηση $f : E \rightarrow R_{\geq 0}$ που ικανοποιεί τις παρακάτω ιδιότητες:

1. Συνθήκη χωρητικότητας: για κάθε ακμή e, ισχύει $0 \leq f_e \leq c_e$, δηλαδή η ροή στην ακμή e είναι το πολύ όσο είναι η χωρητικότητά της.

2. Συνθήκη διατήρησης (conservation conditions): για κάθε κόμβο v ϵ V \ {s, t}, ισχύει ότι $\sum_{e \cdot into \cdot v} f_e = \sum_{e \cdot outof \cdot v} f_e$, δηλαδή η ροή που εισέρχεται σε έναν κόμβο v θα πρέπει να ισούται με την ροή που εξέρχεται από τον κόμβο v.

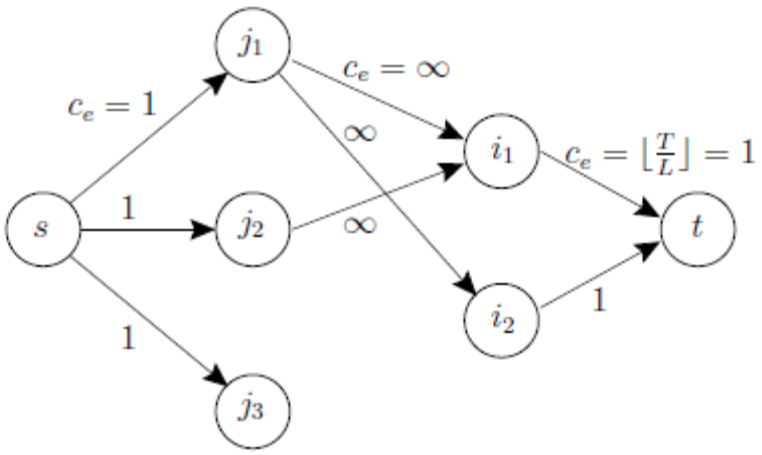
Η αποτίμηση (value) μιας ροής είναι η συνολική ροή που εξέρχεται από τον κόμβο έναρξης s, ή ισάξια η συνολική ροή που εισέρχεται στον κόμβο προορισμού t. Η μέγιστη ροή (max-flow) είναι η ροή με τη μέγιστη αποτίμηση.

Ένα βασικό στοιχείο του αλγόριθμου είναι ότι, με βάση ένα ακέραιο όριο T (integer threshold), που είναι το μέγιστο φορτίο διεργασιών που μπορεί να αναλάβει μια μηχανή, υπολογίζει μια ακέραια μερική ανάθεση x διεργασιών σε μηχανές έτσι ώστε:

i. Κάθε διεργασία να ανατίθεται σε μια low μηχανή ή καθόλου
ii. Το φορτίο σε οποιαδήποτε μηχανή να είναι το πολύ T

iii. Να ανατίθενται ο μέγιστος αριθμός διεργασιών

Μια τέτοια ανάθεση μπορεί να υπολογιστεί με την εύρεση μιας μέγιστης ροής στο δίκτυο. Κατασκευάζουμε ένα κατευθυνόμενο διμερές γράφημα (βλ. σχήμα 5.4) με έναν κόμβο για κάθε διεργασία j και κάθε μηχανή i, και μια ακμή (j, i) άπειρης χωρητικότητας αν η διεργασία j εκτελείται στη μηχανή i ως low, p_{ij} = L. Προσθέτουμε έναν κόμβο έναρξης s με ακμές (s, j) χωρητικότητας 1, και έναν κόμβο προορισμού t με ακμές (i, t) που έχουν χωρητικότητα $\lfloor T / L \rfloor$. Στο σχήμα 5.5 απεικονίζεται ένα παράδειγμα με τρεις διεργασίες και δύο μηχανές, με $p_{i_1 j_1} = p_{i_1 j_2} = p_{i_2 j_1} = L$ και $p_{i_1 j_3} = p_{i_2 j_2} = p_{i_2 j_3} = H$ και T = L.

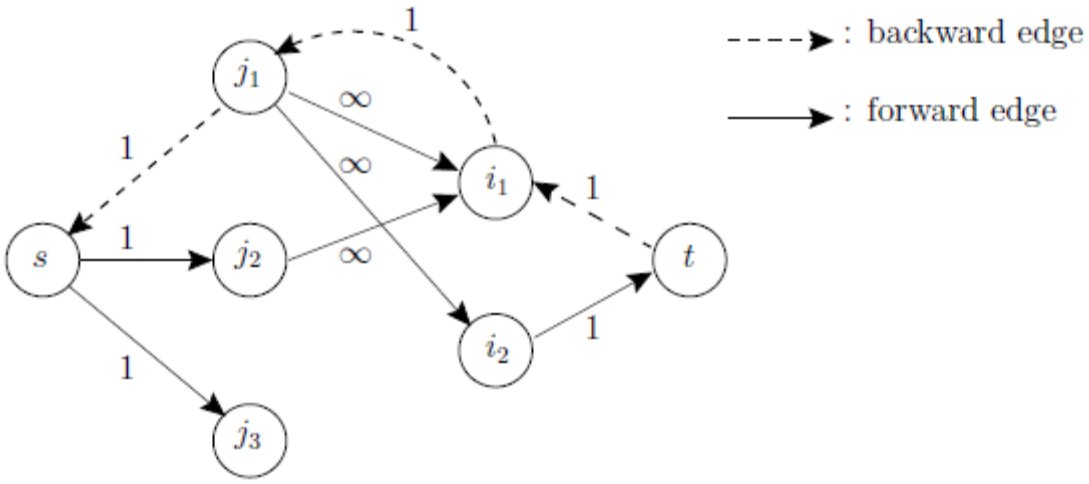


Σχήμα 5.5: Δίκτυο Ροής (p, T), με 3 Διεργασίες, 2 Μηχανές και T=L

Οποιαδήποτε ακέραια ροή σε αυτό το δίκτυο αντιστοιχίζεται σε μια ακέραια μερική ανάθεση x, όπου x_{ij} = 1 αν και μόνο αν υπάρχει ροή ίση με 1 στην ακμή από το j στο i. Ουσιαστικά, το φορτίο σε κάθε μηχανή i, λόγω της ανάθεσης x, είναι L φορές η ροή της ακμής (i, t) κι επομένως είναι το πολύ T, και ο αριθμός των διεργασιών που ανατίθενται με τη x ισούται με την αποτίμηση της ροής. Αντιστρόφως, οποιαδήποτε (μερική) ανάθεση ικανοποιεί τα (a) και (b) απεικονίζεται ως μια ακέραια ροή, όπως αυτή του παραπάνω γραφήματος, με αποτίμηση ίση με τον αριθμό των διεργασιών που

ανατίθενται από τη x. Έτσι, μια ακέραια μέγιστη ροή παράγει την επιθυμητή ανάθεση που μεγιστοποιεί τον αριθμό των ανατεθειμένων διεργασιών. Άρα, οι όροι ανάθεση και ροή θα χρησιμοποιούντα εναλλάξ. Το παραπάνω δίκτυο ροής είναι για (p, T).

Το **γράφημα καταλοίπων (residual graph)** του G, μιας δεδομένης ροής f, έχει τους ίδιους κόμβους που έχει το G και περαιτέρω περιγράφεται ως εξής: Για κάθε αρχική ακμή e ϵ E, α) αν $f_e < c_e$, προσθέτουμε την ακμή e στο γράφημα καταλοίπων με χωρητικότητα $c_e - f_e$, μια τέτοια ακμή είναι κατευθυνόμενη προς τα εμπρός (forward edge) και β) αν $f_e > 0$, προσθέτουμε μια ακμή e΄ στο γράφημα καταλοίπων, που είναι κατευθυνόμενη προς τα πίσω (backward edge), με χωρητικότητα f_e. Το σχήμα 5.6 απεικονίζει το γράφημα καταλοίπων του δικτύου ροής του σχήματος 5.5 όσον αφορά τη μοναδιαία ροή $f_{s,j_1} = f_{j_1,i_1} = f_{i_1,t} = 1$ (και $f_e = 0$ για τις ακμές που απομένουν).



Σχήμα 5.6: Γράφημα καταλοίπων του δικτύου του σχ. 5.5 σε σχέση με τη ροή $j_1 \mapsto i_1$

Αν μια ροή δεν είναι μέγιστη, τότε υπάρχει ένα μονοπάτι στο γράφημα καταλοίπων, που ονομάζεται αυξητικό **μονοπάτι (augmenting path)**, και μπορεί να χρησιμοποιηθεί για να αυξήσει την αποτίμηση της ροής.

Έστω x μια ακέραια ροή που δεν είναι μέγιστη για το (p, T). Τότε υπάρχει ένα αυξητικό μονοπάτι P = (s, j_1, i_1, j_2, i_2, . . . , j_K, i_K, t) στο γράφημα καταλοίπων. Οι ακμές (s, j_1), (j_l, i_l) για l = 1, . . . ,K, και (i_K, t) είναι κατευθυνόμενες προς τα εμπρός, και οι ακμές (i_l, j_{l+1}) για l = 1, . . . ,K-1, είναι κατευθυνόμενες προς τα πίσω. Αυτό το αυξητικό μονοπάτι δείχνει ότι στην τρέχουσα ανάθεση x, η διεργασία j_1 δεν είναι ανατεθειμένη, $x_{(i_l, j_l)}$= 0 πράγμα που προκύπτει από τις ακμές που κατευθύνονται προς τα εμπρός (j_l, i_l), και οι διεργασίες j_2, . . . , j_K ανατίθενται στις μηχανές i_1, . . . , i_{K-1} αντίστοιχα (δηλαδή $x_{(i_l, j_{l+1})}$= 1), πράγμα που προκύπτει από τις ακμές που κατευθύνονται προς τα πίσω (i_l, j_{l+1}). Αυξάνοντας την ανάθεση x χρησιμοποιώντας το μονοπάτι P σημαίνει ότι αλλάζουμε τη ροή/ανάθεση έτσι ώστε στη νέα ανάθεση κάθε διεργασία j_l να ανατίθεται στη μηχανή i_l. Με αυτόν τον τρόπο ανατίθεται πλέον και η διεργασία j_1 αυξάνοντας την αποτίμηση της ροής κατά 1. Παρατηρούμε ότι οποιοδήποτε απλό αυξητικό μονοπάτι δε μειώνει το φορτίο καμιάς μηχανής.

Στο παράδειγμά μας, στην εικόνα 2, ένα πιθανό αυξητικό μονοπάτι είναι το (s, j_2, i_1, j_1, i_2, t). Αυτό μετατρέπει την αρχική ανάθεση (η διεργασία j_1 στη μηχανή i_1) στη νέα ανάθεση: η διεργασία j_1 στη i_2 και η j_2 στη i_1.

Η ύπαρξη αυξητικών μονοπατιών οδηγεί σε έναν απλό αλγόριθμο για τον υπολογισμό μιας μέγιστης ροής, που ονομάζεται Ford-Fulkerson αλγόριθμος (Ford και Fulkerson 1956). Ξεκινάμε με μηδενική ροή (f_e = 0 για όλες τις ακμές), και επαναληπτικά χρησιμοποιούμε αυξητικά μονοπάτια για να αυξήσουμε την αποτίμηση της ροής μέχρι να μην υπάρχει κανένα τέτοιο μονοπάτι. Από τον επαναληπτικό αλγόριθμο συνεπάγεται ότι πάντα υπάρχει μια ακέραια μέγιστη ροή αν όλες οι χωρητικότητες είναι ακέραιοι αριθμοί. Για παράδειγμα, στο δίκτυο ροής (p, T), αυτό τηρείται: ξεκινάμε από μια ακέραια ροή (τη μηδενική), και κάθε φορά που χρησιμοποιούμε ένα αυξητικό μονοπάτι για να αυξήσουμε μια ακέραια ροή, παίρνουμε μια καινούρια ακέραια ροή.

Πρέπει να σημειωθεί ότι είναι πιθανό να προκύψουν πολλές μέγιστες ροές, αλλά εμείς ενδιαφερόμαστε για τις πιο ¨ισοζυγισμένες¨ (balanced) από αυτές. Παίρνουμε μια μέγιστη ροή. Έστω ότι $n^i{}_{p,T}$ είναι η ποσότητα της ροής στην ακμή (i, t) (ή ο αριθμός των διεργασιών που ανατίθενται στην i στη συγκεκριμένη ανάθεση), και έστω $n_{p,T}$ η συνολική αποτίμηση της μέγιστης ροής, τότε $n_{p,T} = \sum_i n^i{}_{p,T}$. Για οποιοδήποτε T΄ ≤ T, ορίζεται το μέγεθος $n^i{}_{p,T/T'} = \min(n^i{}_{p,T}, \lfloor T'/L \rfloor)$, που σημαίνει ότι περικόπτεται η ροή/ανάθεση στην i τόσο ώστε το συνολικό φορτίο στην i να είναι το πολύ T΄. Επίσης, ορίζεται το μέγεθος $n_{p,T/T'} = \sum_i n^i{}_{p,T/T'}$. Στη συνέχεια, ορίζουμε την prefix-maximal ροή ή ανάθεση για το T.

ΟΡΙΣΜΟΣ 5.12 [Prefix-maximal Ροή]: Μια ροή για το δίκτυο (p, T), με άνω φράγμα T, είναι prefix-maximal αν για κάθε ακέραιο T΄ ≤ T, έχουμε $n_{p,T/T'} = n_{p,T'}$.

Αυτό σημαίνει ότι σε μια prefix-maximal ροή για το (p, T), αν περικόψουμε τη ροή σε κάποιο T΄ ≤ T, καταλήγουμε σε μια μέγιστη ροή για το (p, T΄). Μια prefix-maximal ροή για το φράγμα T πάντα υπάρχει, όπως αποδεικνύεται από τον ακόλουθο αλγόριθμο: υπολόγισε μια μέγιστη ροή για το φράγμα T = 1, χρησιμοποίησε απλά αυξητικά μονοπάτια για να αποκτήσεις μια μέγιστη ροή για το φράγμα T = 2, και επανέλαβε, αυξάνοντας κάθε φορά τη μέγιστη ροή για το προηγούμενο φράγμα T = t σε μια μέγιστη ροή για το φράγμα T = t + 1 χρησιμοποιώντας απλά αυξητικά μονοπάτια. Από τη στιγμή που σε κάθε τέτοια μετάβαση, το φορτίο σε κάθε μηχανή δε μειώνεται ποτέ, αυτό καταλήγει σε μια prefix-maximal ροή.

5.5. Κυκλικά Μονότονος Προσεγγιστικός Μηχανισμός

Ο κυκλικά μονότονος προσεγγιστικός μηχανισμός αποτελείται από τρία βήματα και περιγράφεται παρακάτω.

ΑΛΓΟΡΙΘΜΟΣ 2 [LaSw07]**:** Δεδομένου ενός διανύσματος χρόνων επεξεργασίας p, κάνε μια ανάθεση διεργασιών σε μηχανές ως εξής:

1. Υπολόγισε το T*(p) = min{T ≥ H, το T πολλαπλάσιο του L: $n_{p,T} \cdot L + (n - n_{p,T}) \cdot H \leq m \cdot T\}$[11]. Παρατηρούμε ότι $n_{p,T} \cdot L + (n - n_{p,T}) \cdot H - m \cdot T$ είναι φθίνουσα συνάρτηση του T, επομένως το T*(p) μπορεί να υπολογιστεί σε πολυωνυμικό χρόνο μέσω δυαδικής αναζήτησης. Ο σκοπός, σε αυτό το βήμα, είναι να υπολογιστεί ένας αρκετά μεγάλος αριθμός (m·T) ως άνω φράγμα T*(p), έτσι ώστε αρχικά να ανατεθεί ο μέγιστος αριθμός low διεργασιών που είναι δυνατό να ανατεθεί μέχρι αυτό το άνω φράγμα, και στη συνέχεια να έχει απομείνει αρκετός ελεύθερος χώρος σε όλες τις μηχανές, ώστε να ανατεθούν κλασματικά και οι high διεργασίες που έχουν απομείνει.

2. Υπολόγισε μια prefix-maximal ροή για το άνω φράγμα T*(p) και την αντίστοιχη μερική ανάθεση (για παράδειγμα, η διεργασίας j ανατίθεται στη μηχανή i αν και μόνο αν υπάρχει μια μονάδα ροής στην ακμή (j, i)). Σε αυτό το βήμα, γίνεται η ανάθεση του μέγιστου αριθμού low διεργασιών.

3. Ανέθεσε τις διεργασίες που έχουν απομείνει, δηλαδή τις διεργασίες που δεν έχουν ανατεθεί στην προηγούμενη φάση και άρα είναι high, με άπληστο τρόπο ως εξής: τοποθέτησε αυτές τις διεργασίες σε μια τυχαία σειρά και επαναληπτικά ανέθεσε κάθε διεργασία στη μηχανή με το τρέχον μικρότερο φορτίο (το φορτίο που υπολογίζεται περιλαμβάνει και τις διεργασίες που ανατέθηκαν στην προηγούμενη φάση).

Ο αλγόριθμος απαιτεί τον υπολογισμό μιας prefix-maximal ανάθεσης για το άνω φράγμα T*(p). Η απόδειξη της ύπαρξης της prefix-maximal ροής, που περιγράφθηκε

[11] Ο περιορισμός το T να είναι πολλαπλάσιο του L δεν είναι απαραίτητος, αλλά απλοποιεί τη διαδικασία.

παραπάνω, παράγει έναν αλγόριθμο για τον υπολογισμό της, που χρησιμοποιεί συνολικά το πολύ n αυξητικά μονοπάτια, στην περίπτωση που υπάρχουν n διεργασίες για να ανατεθούν (επομένως η αποτίμηση της μέγιστης ροής είναι το πολύ n, για οποιοδήποτε άνω φράγμα T). Αυτός ο αλγόριθμος είναι πολυωνυμικού χρόνου.

ΘΕΩΡΗΜΑ 5.13: Ο Αλγόριθμος 2 είναι ένας πολυωνυμικού χρόνου ντετερμινιστικός 2-προσεγγιστικός αλγόριθμος, για το πρόβλημα της ανάθεσης διεργασιών στην περίπτωση ¨δύο-τιμών¨, και είναι υλοποιήσιμος. Επιπλέον, οι αποζημιώσεις που παράγουν έναν φιλαλήθη μηχανισμό μπορούν να υπολογιστούν αποδοτικά.

Στις επόμενες παραγράφους αποδεικνύεται το Θεώρημα 5.13.

5.5.1. Ο Αλγόριθμος 2 είναι 2-προσεγγιστικός

Υπενθυμίζουμε ότι το OPT(p) δείχνει το βέλτιστο makespan από τις ακέραιες αναθέσεις που προκύπτουν για το p.

ΙΣΧΥΡΙΣΜΟΣ 5.14: Αν OPT(p) < H, το makespan είναι το πολύ OPT(p).

Απόδειξη: Αν OPT(p) < H, τότε η βέλτιστη ανάθεση αναθέτει όλες τις διεργασίες σε low μηχανές, επομένως $n_{p,OPT(p)}$ = n.

Στον Αλγόριθμο 2 θέτουμε το T*(p) = L $\cdot \left\lceil \frac{H}{L} \right\rceil$, δηλαδή ορίζουμε έναν μεγάλο αριθμό, H, να είναι το άνω φράγμα του φορτίου που μπορεί να επεξεργαστεί η μηχανή. Επιπλέον, αφού στο βήμα 2 υπολογίστηκε μια prefix-maximal ροή για το άνω φράγμα T*(p), αν περικόψουμε το T*(p) σε OPT(p), έχουμε $n_{p,T^*(p)/OPT(p)} = n_{p,OPT(p)}$ = n, που σημαίνει ότι το φορτίο σε κάθε μηχανή είναι το πολύ OPT(p). Επομένως, το makespan

είναι το πολύ (και άρα ακριβώς) OPT(p). Από τα παραπάνω προκύπτει ότι, τόσο η δική μας ανάθεση όσο και η βέλτιστη αναθέτουν όλες τις διεργασίες σε low μηχανές.

ΙΣΧΥΡΙΣΜΟΣ 5.15: Αν $OPT(p) \geq H$, τότε $T^*(p) \leq L \cdot \left\lceil \frac{OPT(p)}{L} \right\rceil \leq OPT(p) + L$

Απόδειξη: Έστω $n_{OPT(p)}$ ο αριθμός των διεργασιών που ανατίθενται σε low μηχανές σε μια βέλτιστη ανάθεση. Το συνολικό φορτίο σε όλες τις μηχανές σε αυτή τη βέλτιστη ανάθεση είναι ακριβώς $n_{OPT(p)} \cdot L + (n - n_{OPT(p)}) \cdot H$, και είναι το πολύ $m \cdot OPT(p)$, αφού κάθε μηχανή έχει φορτίο το πολύ OPT(p).

Αν πάρουμε το $T = L \cdot \left\lceil \frac{OPT(p)}{L} \right\rceil \geq OPT(P) \geq H$ *σαν υποψήφιο* άνω *φράγμα*, τότε $n_{p,T} \geq n_{OPT(p)}$, δηλαδή, *το* $n_{p,T}$ είναι τουλάχιστον ο αριθμός των διεργασιών που ανατέθηκαν σε low μηχανές σε μια βέλτιστη ανάθεση (αφού $T \geq OPT(P)$). Έχουμε ότι

$n_{p,T} \cdot L + (n - n_{p,T}) \cdot H = n_{p,T} \cdot L + n \cdot H - n_{p,T} \cdot H$

$= n \cdot H - (H - L)\, n_{p,T} \leq n \cdot H - (H - L)\, n_{OPT(p)}$

$= n_{OPT(p)} \cdot L + (n - n_{OPT(p)}) \cdot H \leq m \cdot OPT(p) \leq m \cdot T.$

Επομένως, το συνολικό φορτίο, αφού ανατεθούν και οι high διεργασίες που είχαν απομείνει είναι το πολύ $m \cdot OPT(p) \leq m \cdot T$. Στον Αλγόριθμο 2, το $T^*(p)$ είναι το μικρότερο από αυτά τα T, κι επομένως είναι το πολύ $T^*(p) \leq T = L \cdot \left\lceil \frac{OPT(p)}{L} \right\rceil$.

ΙΣΧΥΡΙΣΜΟΣ 5.16: Κάθε διεργασία που ανατίθεται στο βήμα 3 του αλγορίθμου ανατίθεται σε high μηχανή.

Απόδειξη: Έστω ότι η διεργασία j ανατίθεται στη μηχανή i στο βήμα 3. Υπενθυμίζουμε ότι στο βήμα 2 ανατίθεται ο μέγιστος αριθμός low διεργασιών που είναι δυνατό να ανατεθεί μέχρι το άνω φράγμα $T^*(p)$. Αν p_{ij} = L, τότε πρέπει να έχουμε $n^i_{p,T^*(p)} = T^*(p)/L$, αλλιώς η διεργασία j θα μπορούσε να είχε ανατεθεί στη μηχανή i στο βήμα 2 για να αποκτήσουμε μια ροή με μεγαλύτερη αποτίμηση. Επομένως, στο σημείο πριν ακριβώς ανατεθεί η low διεργασία j στο βήμα 3, το φορτίο κάθε μηχανής πρέπει να είναι τουλάχιστον T*(p). Το συνολικό φορτίο αφού ανατεθεί η j θα είναι τουλάχιστον m · T*(p) + L > m · T*(p). Αλλά το συνολικό φορτίο είναι επίσης το πολύ $n_{p,T^*(p)} \cdot L + (n - n_{p,T^*(p)}) \cdot H \le m \cdot T^*(p)$, κι έτσι έχουμε αντίφαση. Άρα, κάθε διεργασία που ανατίθεται στο βήμα 3 του αλγορίθμου ανατίθεται σε high μηχανή.

ΛΗΜΜΑ 5.17: Ο Αλγόριθμος 2 επιστρέφει μια ανάθεση με makespan το πολύ OPT(p) + max{L, H(1-1/m)} ≤ 2· OPT(p).

Απόδειξη: Αν OPT(p) < H, τότε από τον ισχυρισμό 5.16, προκύπτει ότι το λήμμα ισχύει. Υποθέτουμε ότι OPT(p) ≥ H, από τον ισχυρισμό 5.15, ξέρουμε ότι T*(p) ≤ OPT(p) + L. Αν δεν υπάρχουν μη ανατεθειμένες διεργασίες μετά το βήμα 2 του αλγορίθμου, τότε το makespan είναι το πολύ T*(p) που σημαίνει ότι το λήμμα ισχύει.

Υποθέτουμε ότι υπάρχουν μη ανατεθειμένες διεργασίες μετά το βήμα 2 του αλγορίθμου. Για να ολοκληρωθεί η απόδειξη, θα δείξουμε ότι το makespan μετά το βήμα 3 είναι το πολύ T + H (1-1/m), όπου T = min{T*(p), OPT(p)}. Υποθέτουμε ότι αυτό είναι λάθος.

Έστω ότι i είναι η μηχανή με το μέγιστο φορτίο, δηλαδή l_i > T + H (1-1/m). Έστω j η διεργασία που ανατίθεται τελευταία στη μηχανή i στο βήμα 3. Στο σημείο πριν ακριβώς ανατεθεί η διεργασία στην i έχουμε l_i > T− H/m. Επιπλέον, αφού η j ανατίθεται στην i, σύμφωνα με τον κανόνα του αλγορίθμου, το φορτίο σε όλες τις άλλες μηχανές πρέπει να είναι τουλάχιστον l_i. Έτσι, το συνολικό φορτίο αφού ανατεθεί η j είναι τουλάχιστον H + m· l_i > m · T (p_{ij} = H από τον ισχυρισμό 5.16).

Επίσης, για οποιαδήποτε ανάθεση διεργασιών στις μηχανές στο βήμα 3, το συνολικό φορτίο είναι το πολύ $n_{p,T^*(p)} \cdot L + (n - n_{p,T^*(p)}) \cdot H$ δεδομένου ότι $n_{p,T^*(p)}$ διεργασίες ανατέθηκαν σε low μηχανές. Συνδυάζοντας τα παραπάνω, θα πρέπει να έχουμε $m \cdot T < n_{p,T^*(p)} \cdot L + (n - n_{p,T^*(p)}) \cdot H$. Όμως, γνωρίζουμε ότι $m \cdot T \geq n_{p,T^*(p)} \cdot L + (n - n_{p,T^*(p)})$, και άρα έχουμε αντίφαση, που σημαίνει ότι το makespan μετά το βήμα 3 είναι το πολύ $T + H (1-1/m)$, όπου $T = \min\{T^*(p), OPT(p)\}$.

Αν $T = T^*(p)$, το λήμμα ισχύει από τον ορισμό του $T^*(p)$. Αν $T = OPT(p)$, τότε αν $n_{OPT(p)}$ ο αριθμός των διεργασιών που ανατίθενται σε low μηχανές σε μια βέλτιστη ανάθεση, έχουμε ότι $n_{p,T^*(p)} \geq n_{OPT(p)}$, και άρα, $n_{p,T^*(p)} \cdot L + (n - n_{p,T^*(p)}) \cdot H \leq n_{OPT(p)} \cdot L + (n - n_{OPT(p)}) \cdot H$. Αυτό ακριβώς είναι το συνολικό φορτίο σε μια βέλτιστη ανάθεση, και είναι το πολύ $m \cdot OPT(p)$.

5.5.2. Ο Αλγόριθμος 2 είναι Κυκλικά Μονοτονικός

Η συνθήκη που ισχύει όταν έχουμε κυκλική μονοτονικότητα είναι $\sum_{k=1}^{K} \sum_j x_{ij}^{k+1} (p_{ij}^{k} - p_{ij}^{k+1}) \geq 0$ (πόρισμα 5.5). Στην ειδική περίπτωση του προβλήματος ανάθεσης διεργασιών των ¨δύο-τιμών¨ {L, H} που μελετάμε, ορίζουμε μια μηχανή i κι έναν δείκτη k για κάθε μη μηδενικό όρο $\pm(H - L)$ του τύπου της παραπάνω συνθήκης. Αντικαθιστούμε τη συνθήκη, με μια ισότιμη και πιο βολική συνθήκη, που χαρακτηρίζεται από τον αριθμό των διεργασιών που ανατίθενται στη μηχανή i στο x^k, οι οποίες μεταβάλλονται από high σε low, και αντίστροφα, όταν ο παίκτης από τη δήλωση p^k μετακινείται στην p^{k+1}.

Ορίζουμε τους αριθμούς :

$$n_H^{k+1,k} = \left| \{ j : x_{ij}^{k+1} = 1, p_{ij}^{k+1} = L, p_{ij}^{k} = H \} \right| \qquad \text{(σχέση 5.4)}$$

Ο παίκτης από την κατάσταση k μετακινείται στην κατάσταση k+1 και δηλώνει την ψευδή τιμή L για το χρόνο επεξεργασίας του (όρος H-L).

$$n_L^{k+1,k} = \left|\{j : x_{ij}^{k+1} = 1, p_{ij}^{k+1} = H, p_{ij}^{k} = L\}\right| \qquad (5.5)$$

Ο παίκτης από την κατάσταση k μετακινείται στην κατάσταση k+1 και δηλώνει την ψευδή τιμή H για το χρόνο επεξεργασίας του (όρος L- H).

Τότε, $\sum_{k=1}^{K} \sum_j x_{ij}^{k+1}(p_{ij}^{k} - p_{ij}^{k+1}) \geq 0 \Rightarrow \sum_{k=1}^{K}[n_H^{k+1,k}(H-L) + n_L^{k+1,k}(L-H)] \geq 0 \Rightarrow$

$\sum_{k=1}^{K}[(n_H^{k+1,k} - n_L^{k+1,k})(H-L)] \geq 0$ και αφού H-L ≥ 0 έχουμε ότι

$\sum_{k=1}^{K}(n_H^{k+1,k} - n_L^{k+1,k}) \geq 0$

ΠΡΟΤΑΣΗ 5.18: Η κυκλική μονοτονικότητα στην ειδική περίπτωση του προβλήματος ανάθεσης διεργασιών των ¨δύο-τιμών¨ {L, H} είναι ισοδύναμη με τη συνθήκη: για κάθε παίκτη i, κάθε p_{-i}, κάθε ακέραιο K, και κάθε $p_i^k, ..., p_i^K$

$$\sum_{k=1}^{K}(n_H^{k+1,k} - n_L^{k+1,k}) \geq 0 \qquad (5.6)$$

Δηλαδή, όταν έχουμε μετάβαση από την ανάθεση k στην ανάθεση k+1, η διαφορά $n_H^{k+1,k} - n_L^{k+1,k}$ (αριθμός διεργασιών, που ανατίθεται στην i, και μειώθηκε ο χρόνος επεξεργασίας τους από H σε L μείον τον αριθμό διεργασιών, που ανατίθεται στην i, και αυξήθηκε ο χρόνος επεξεργασίας τους από L σε H) θα πρέπει να αυξάνεται.

Διαισθητικά, ο Αλγόριθμος 2 ικανοποιεί τη συνθήκη 5.6: Θεωρούμε την ειδική περίπτωση όπου όλα τα άνω όρια $T^*(p_i^k, p_{-i})$ για k =1,…, K είναι ίσα, και συμβολίζονται με T*. Έστω ότι με p^k συμβολίζεται η είσοδος (p_i^k, p_{-i}). Θεωρούμε την ανάθεση x^2 που υπολογίζεται για την είσοδο p^2, και η οποία αναθέτει σε low μηχανές ακριβώς n_{p^2,T^*} αριθμό διεργασιών. Ισχυριζόμαστε ότι αν αναθέσουμε στη μηχανή i όλες τις j διεργασίες για τις οποίες $x_{ij}^2 = 1$ και $p_{ij}^1 = L$, τότε αποκτούμε μια έγκυρη ροή για (p^1, T^*) με αποτίμηση $n_{p^2,T^*} - n_H^{2,1} + n_L^{2,1}$. Το σύνολο των low διεργασιών στη μηχανή i μπορεί να γραφτεί ως εξής:

$$\{j: x_{ij}^2 = 1,\ p_{ij}^2 = L\} \setminus \{j: x_{ij}^2 = 1,\ p_{ij}^2 = L,\ p_{ij}^1 = H\} \cup \{j: x_{ij}^2 = 1,\ p_{ij}^2 = H,\ p_{ij}^1 = L\}$$

Δηλαδή, στη ροή (p^1, T^*) ο αριθμός των low διεργασιών είναι: όλες οι L διεργασίες της ανάθεσης x^2 ($p_{ij}^2 = L$) μείον όσες από αυτές έγιναν H στην ανάθεση x^1 συν όσες διεργασίες από H έγιναν L στην ανάθεση x^1. Το low φορτίο στην i είναι το πολύ T*:

$$\text{Low-Load on i} = T^* - n_H^{2,1} \cdot L + n_L^{2,1} \cdot L = T^* - \underbrace{(n_H^{2,1} - n_L^{2,1}) \cdot L}_{\geq 0} \leq T^*$$

Αφού n_{p^1,T^*} είναι η αποτίμηση της μέγιστης ροής για το (p^1, T^*) έχουμε ότι $n_{p^1,T^*} \geq n_{p^2,T^*} - n_H^{2,1} + n_L^{2,1}$. Όλα τα παραπάνω ισχύουν για οποιεσδήποτε αναθέσεις p^{k+1} και p^k κι επομένως ισχύει η ανισότητα $n_{p^k,T^*} \geq n_{p^{k+1},T^*} - n_H^{k+1,k} + n_L^{k+1,k}$ (όπου $K+1 \equiv 1$). Προσθέτοντας τις για όλα τα k καταλήγουμε στη συνθήκη 5.6:

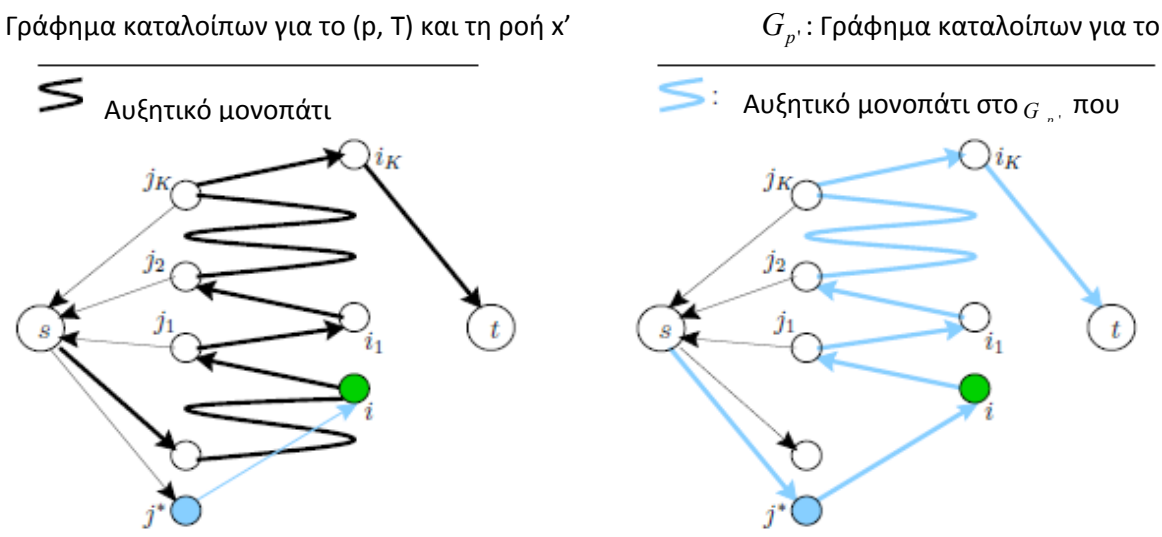
$$\sum_{k=1}^{K} n_{p^k,T^*} \geq \sum_{k=1}^{K} n_{p^{k+1},T^*} - \sum_{k=1}^{K} n_H{}^{k+1,k} + \sum_{k=1}^{K} n_L{}^{k+1,k} \Rightarrow$$

$$\sum_{k=1}^{K} (n_H{}^{k+1,k} - n_L{}^{k+1,k}) \geq \sum_{k=1}^{K} n_{p^{k+1},T^*} - \sum_{k=1}^{K} n_{p^k,T^*} \geq \sum_{k=1}^{K} n_{p^k,T^*} - \sum_{k=1}^{K} n_{p^k,T^*} \Rightarrow$$

$$\sum_{k=1}^{K} (n_H{}^{k+1,k} - n_L{}^{k+1,k}) \geq 0$$

Έως τώρα το κοινό άνω φράγμα T* βοήθησε στο να συγκρίνουμε τις διαφορετικές αναθέσεις p^k . Στην παρακάτω ανάλυση αντικαθιστούμε το T* με το άνω φράγμα T^L = T*($\vec{L}$, p_{-i}), όπου $\vec{L}$ είναι το διάνυσμα με όλους τους low χρόνους επεξεργασίας. Πιο συγκεκριμένα, θα αποδείξουμε τη συνθήκη 5.6 αποδεικνύοντας ότι η ανισότητα $n_{p^1,T^L} \geq n_{p^2,T^L} - n_H{}^{2,1} + n_L{}^{2,1}$ L ισχύει για οποιοδήποτε $p^1=(p_i{}^1, p_{-i})$ *και* $p^2=(p_i{}^2, p_{-i})$.

ΛΗΜΜΑ 5.19: Θεωρούμε οποιαδήποτε δύο στιγμιότυπα p = (p_i, p_{-i}) και p' = (p_i', p_{-i}), όπου $p' \geq$ p, δηλαδή $p_{ij}' \geq p_{ij}$ $\forall$ j. Αν T είναι ένα άνω φράγμα έτσι ώστε $n_{p,T} > n_{p',T}$, τότε κάθε μέγιστη ροή x΄ για το δίκτυο (p', T) πρέπει να αναθέτει όλες τις διεργασίες j έτσι ώστε p_{ij}' = L.



Σχήμα 5.7: Γραφήματα Καταλοίπων

Απόδειξη: Έστω ότι $G_{p'}$ το γράφημα καταλοίπων για το *(p', T) και τη ροή* x' (βλ. σχήμα 5.7). Για να καταλήξουμε σε αντίφαση υποθέτουμε ότι υπάρχει μια διεργασία j* με $p_{ij^*}' = L$ που δεν είναι ανατεθειμένη στο x'. Αφού $p_i' \geq p_i$, όλες οι ακμές (j, i) που παρουσιάζονται στο δίκτυο (p΄, T) παρουσιάζονται επίσης και στο δίκτυο (p, T). Έτσι, η x΄ είναι μια έγκυρη ροή για το (p, T). Αλλά δεν είναι μέγιστη ροή, αφού $n_{p,T} > n_{p',T}$. Επομένως, υπάρχει αυξητικό μονοπάτι P στο γράφημα καταλοίπων για το (p, T) και τη ροή x΄.

Παρατηρούμε ότι ο κόμβος i θα πρέπει οπωσδήποτε να περιλαμβάνεται στο P, αλλιώς το P θα ήταν αυξητικό μονοπάτι και στο γράφημα καταλοίπων $G_{p'}$, πράγμα που έρχεται σε αντίφαση με το γεγονός ότι η x΄ είναι μέγιστη ροή για το δίκτυο (p', T), και άρα δεν έχει αυξητικά μονοπάτια. Αυτό σημαίνει ότι υπάρχει ένα μονοπάτι (όχι αυξητικό) $P' \subset P$ από τον κόμβο i στον κόμβο προορισμού t. Let P΄ = (i, j_1, i_1, . . . , j_K, i_K, t). Όλες οι ακμές από το P΄ παρουσιάζονται επίσης σαν ακμές στο $G_{p'}$: όλες οι κατευθυνόμενες προς τα πίσω ακμές (i_l, j_{l+1}) παρουσιάζονται επειδή μια τέτοια ακμή σημαίνει ότι έχουμε ανάθεση διεργασίας $x'_{i_l j_{l+1}}=1$ (η μηχανή i_k δεν εκτελεί καμία διεργασία) και οι κατευθυνόμενες προς τα εμπρός ακμές (j_l, i_l), που παρουσιάζονται για $i_l \neq i$, ώστε $p'_{i_l j_l} = p_{i_l j_l} = L$, και $x'_{i_l j_l} = 0$ (που σημαίνει ότι η διεργασία j* δεν είναι ανατεθειμένη). Αλλά τότε υπάρχει ένα αυξητικό μονοπάτι (s, j*, i, j_1, i_1, . . . , j_K, i_K, t) στο $G_{p'}$ (βλ. σχήμα 5.7) που έρχεται σε αντίφαση με την υπόθεση για την ύπαρξη μέγιστης ροής στην ανάθεση x΄.

Έστω $\vec{L}$ το διάνυσμα των χρόνων επεξεργασίας, που δείχνει ότι όλες οι διεργασίες εκτελούνται σαν low. Για την είσοδο p = ($\vec{L}$, p_{-i}), ορίζουμε το $T_i^L(p_{-i}) = T^*(\vec{L}, p_{-i})$. Αφού μελετάμε μια μηχανή i και διατηρούμε σταθερά τα p_{-i}, χρησιμοποιούμε τη

συντομογραφία T^L και p^L για τις παραπάνω ποσότητες. Για οποιοδήποτε στιγμιότυπο p = (p_i, p_{-i}) ισχύει ότι $T^*(p) \geq T^L$, αφού στο T^L όλες οι διεργασίες εκτελούνται σαν low.

ΠΟΡΙΣΜΑ 5.20: Έστω p = (p_i, p_{-i}) ένα οποιοδήποτε στιγμιότυπο και έστω x μια οποιαδήποτε prefix-maximal ροή για το (p, T*(p)). Τότε το low φορτίο στη μηχανή i είναι το πολύ T^L.

Απόδειξη: Έστω ότι T* = T*(p). Αν T* = T^L, τότε προφανώς αυτό ισχύει. Αλλιώς, θεωρούμε την ανάθεση x΄ που αποκτήθηκε με την περικοπή του x σε T^L. Αφού η ροή x είναι prefix-maximal, γνωρίζουμε ότι η x΄ συνιστά μια μέγιστη ροή για το (p, T^L). Επίσης, ισχύει *ότι* $n_{p,T^L} < n_{p^L,T^L}$ επειδή T* > T^L (δηλαδή, ο αριθμός των διεργασιών που εκτελούνται είτε σαν L είτε σαν H είναι μικρότερος από τον αριθμό των διεργασιών που εκτελούνται μόνο σαν L). Έτσι, σύμφωνα με το λήμμα 5.8, αυτή η περικομμένη ροή πρέπει να αναθέτει όλες τις low διεργασίες στη μηχανή i. Από τον ορισμό της περικομμένης ροής, έπεται ότι $x_{ij} \geq x_{ij}$ ' για όλα τα i, j, έτσι αν μια διεργασία ανατίθεται από τη x΄ τότε θα ανατίθεται σίγουρα και στη x και μάλιστα θα ανατίθεται στην ίδια μηχανή. Δηλαδή, οι αναθέσεις των low διεργασιών στη μηχανή i παραμένουν ίδιες στη x΄ και x, κι επομένως, το low- φορτίο στη μηχανή i δε μεταβάλλεται όταν μετακινούμαστε από την ανάθεση x΄ στη x. *Άρα, το low-φορτίο στην i (υπό την ανάθεση x) είναι το πολύ* T^L.

ΛΗΜΜΑ 5.21[12]**:** Αν T*(p^1) > T^L, τότε για οποιοδήποτε στιγμιότυπο p^1=(p_i^1, p_{-i}) και p^2=(p_i^2, p_{-i}) ισχύει ότι $n_{p^1,T^L} \geq n_{p^2,T^L} - n_H^{2,1} + n_L^{2,1}$

Απόδειξη: Παίρνουμε την prefix-maximal ροή x^2 για το (p^2, T^2), περικόβουμε το T^2 σε T^L και αφαιρώ από αυτή την ανάθεση όλες τις διεργασίες $n_H^{2,1}$, που είναι όλες οι διεργασίες j για τις οποίες ισχύει x^2_{ij} = 1, p^2_{ij} = L, , p^1_{ij} = H, δηλαδή αφαιρώ όλες τις

[12] Έστω T^1 = T*(p^1) και T^2 = T*(p^2)

διεργασίες που από L έγιναν H. Συμβολίζουμε αυτή τη ροή με x, που είναι έγκυρη ροή για το (p^1, T^L), και η αποτίμηση αυτής της ροής είναι ακριβώς $n_{p^2,T^2/T^L} - n_H{}^{2,1} = n_{p^2,T^L} - n_H{}^{2,1}$ (από τον ορισμό της περικομμένης ροής ισχύει ότι $n_{p^2,T^2/T^L} = n_{p^2,T^L}$)Επίσης, καμία από τις διεργασίες $n_L{}^{2,1}$ που από H στο (p^2, T^2) έγιναν L στο (p^1, T^L) δεν ανατίθεται από την ανάθεση x, αφού κάθε τέτοια διεργασία j είναι high στη μηχανή i στο p^2 (σύμφωνα με το πόρισμα 5.20 ανατίθενται μόνο όλες οι L διεργασίες του (p^2, T^2)).

Αφού $T^1 > T^L$, πρέπει να έχουμε ότι $n_{p^1,T^L} < n_{p^L,T^L}$. Έτσι, αν προσαυξήσουμε τη x σε μια μέγιστη ροή για το (p^1, T^L), σύμφωνα με το λήμμα 5.19 (αν θέσουμε $p = p^L$ και $p' = p^1$), θα πρέπει να αναθέσουμε όλες τις $n_L{}^{2,1}$ διεργασίες σε αυτή τη μέγιστη ροή. Έτσι, η αποτίμηση αυτής της μέγιστης ροής είναι τουλάχιστον (αποτίμηση της x) + $n_L{}^{2,1}$ L, δηλαδή,

$$n_{p^1,T^L} \geq n_{p^2,T^L} - n_H{}^{2,1} + n_L{}^{2,1} \qquad \text{(σχέση 5.7)}$$

ΛΗΜΜΑ 5.22: Αν $T^*(p^1) = T^L$, τότε ισχύει η (5.7)

Απόδειξη: Έστω ότι έχουμε τις ολοκληρωμένες αναθέσεις x^1 και x^2, δηλαδή τις αναθέσεις που προκύπτουν μετά τα βήματα 2 και 3 του αλγόριθμου και για τις εισόδους p^1 και p^2 αντίστοιχα. Έστω S = {j : $x^2_{ij} = 1$ και $p^2_{ij} = L$ } και S΄΄ = {j : $x^2_{ij} = 1$ και $p^1_{ij} = L$ }. Συνεπώς, |S΄΄| = |S| - $n_H{}^{2,1} + n_L{}^{2,1}$ και |S| = $n^i_{p^2,T^2} = n^i_{p^2,T^2/T^L}$ (από το πόρισμα 5.20).

Έστω T΄΄ = |S΄΄|·L.

Έχουμε δύο περιπτώσεις:

- **$T'' \leq T^L$**

Θεωρούμε τη ροή (p^1, T^L): ανέθεσε σε όλες τις μηχανές εκτός από την i τις low-αναθέσεις της ροής x^2 με το περικομμένο T^L, και ανέθεσε τις διεργασίες του συνόλου S΄΄ στη μηχανή i. Αυτή είναι μια έγκυρη ανάθεση για το (p^1, T^L) αφού σε όλες τις μηχανές εκτός από την i έχουμε μέγιστο φορτίο T^L και στη μηχανή i έχουμε $T'' \leq T^L$. Η αποτίμησή του ισούται με $\sum_{i' \neq i} n^{i'}_{p^2,T^2/T^L} + |S''| = n_{p^2,T^2/T^L} - n_H^{2,1} + n_L^{2,1} = n_{p^2,T^L} - n_H^{2,1} + n_L^{2,1}$. Κι επομένως, αφού η αποτίμηση για τη μέγιστη ροή n_{p^1,T^L} δεν είναι μικρότερη έχουμε ότι ισχύει η (5.7).

- **$T'' > T^L \Rightarrow T'' \geq T^L + L$ αφού T'' και T^L είναι πολλαπλάσια του L**

Έχουμε ότι $T'' > T^L$ επειδή $T'' = |S''| \cdot L$ και $|S| \cdot L \leq T^L$ (από το πόρισμα 5.20) προκύπτει ότι $|S''| \cdot L > |S| \cdot L \Rightarrow |S| - n_H^{2,1} + n_L^{2,1} > |S| \Rightarrow n_L^{2,1} > n_H^{2,1} \geq 0$.

Έστω $M = n_{p^2,T^2} - n_H^{2,1} + n_L^{2,1} = \sum_i n^i_{p^2,T^2} - n_H^{2,1} + n_L^{2,1} = \sum_{i' \neq i} n^{i'}_{p^2,T^2} + |S| - n_H^{2,1} + n_L^{2,1} = |S''| + \sum_{i' \neq i} n^{i'}_{p^2,T^2}$

Έστω N οι διεργασίες που ανατίθενται στη μηχανή i στην ανάθεση x^2.
Έχουμε $|S''| \cdot L + (N - |S''|) \cdot H \geq |S''| \cdot L > T^L$ (από την υπόθεση). Στο στιγμιότυπο p^2, θεωρούμε το σημείο, του βήματος 3 του αλγορίθμου, πριν ακριβώς ανατεθεί η τελευταία high διεργασία στη μηχανή i (υπάρχει σίγουρα high διεργασία στο p^2 επειδή ξέρουμε ότι $n_L^{2,1} > 0$). Το φορτίο στην i σε αυτό το σημείο είναι $|S| \cdot L + (N - |S| - 1) \cdot H =$

$(|S''| + n_H^{2,1} - n_L^{2,1}) L + (N - |S| - 1) \cdot H \geq |S''| \cdot L\ (-L + L) - n_L^{2,1} \cdot L + (N - |S| - 1) \cdot H =$

$|S''| \cdot L - L - (n_L^{2,1} - 1) \cdot L + (N - |S| - 1) \cdot H$

που είναι τουλάχιστον $|S''| \cdot L - L \geq T^L$ αφού $n_L^{2,1} - 1 \leq N - |S| - 1$: οι διεργασίες που από Η στο p^2 έγιναν L στο p^1 ($n_L^{2,1}$) είναι λιγότερες ή ίσες από το σύνολο $(N - |S|)$ των Η διεργασιών στο p^2, ουσιαστικά στο σύνολο $(N - |S|)$ περιλαμβάνονται και οι $(n_L^{2,1})$ διεργασίες.

Επειδή, ο αλγόριθμος είναι άπληστος (πριν προχωρήσει σε επόμενη ανάθεση συγκρίνει τα φορτία των μηχανών), κάθε μηχανή $i' \neq i$ έχει τουλάχιστον αυτό το φορτίο σε αυτό το σημείο του αλγορίθμου, έτσι $\sum_j p_{i'j}^2 x_{i'j}^2 \geq T^L$. Προσθέτοντας όλες αυτές τις ανισότητες για όλα τα $i' \neq i$ και την προηγούμενη ανισότητα για το i , παίρνουμε

$$\underbrace{\sum_{i' \neq i} \sum_j p_{i'j}^2 x_{i'j}^2 + |S''| \cdot L + (N - |S''|) \cdot H}_{} > m \cdot T^L$$

$$M \cdot L + (n-M) \cdot H > m \cdot T^L$$

Αφού (σύμφωνα με την υπόθεση) $T^*(p^1) = T^L$, έχουμε

$$n_{p^1,T^L} \cdot L + (n - n_{p^1,T^L}) \cdot H \leq m \cdot T^1 = m \cdot T^L$$

Επομένως,

$M \cdot L + (n-M) \cdot H > n_{p^1,T^L} \cdot L + (n - n_{p^1,T^L}) \cdot H \Rightarrow$

$(M - n_{p^1,T^L}) \cdot L + (n - M - n + n_{p^1,T^L}) \cdot H > 0 \Rightarrow$

$(M - n_{p^1,T^L}) \cdot L + (n_{p^1,T^L} - M) \cdot H > 0 \Rightarrow (n_{p^1,T^L} - M) \cdot (H - L) > 0$

Άρα, πρέπει

n_{p^1,T^L} -M > 0 $\Rightarrow n_{p^1,T^L} > \mathrm{M} = n_{p^2,T^2} - n_H^{2,1} + n_L^{2,1}$.

ΛΗΜΜΑ 5.23: Ο Αλγόριθμος 2 ικανοποιεί την ιδιότητα της κυκλικής μονοτονικότητας

Απόδειξη: Παίρνοντας $p^1 = p^k$ και $p^2 = p^{k+1}$ στην (5.7) έχουμε την ανισότητα $n_{p^k,T^L} \geq n_{p^{k+1},T^L} - n_H^{k+1,k} + n_L^{k+1,k}$. Προσθέτοντας όλες αυτές τις ανισότητες για k=1,…K (όπου K+1 ≡ 1) καταλήγουμε στη συνθήκη 5.6: $\sum_{k=1}^{K}(n_H^{k+1,k} - n_L^{k+1,k}) \geq 0$

5.6. Υπολογισμός Αποζημιώσεων

Σύμφωνα με το πόρισμα 5.6, υπάρχουν αποζημιώσεις που χρησιμοποιεί ένας φιλαλήθης μηχανισμός, ωστόσο αυτές οι αποζημιώσεις θα πρέπει να υπολογίζονται σε πολυωνυμικό χρόνο, προκειμένου να αποκτήσουμε έναν μηχανισμό πολυωνυμικού χρόνου. Δεν είναι σίγουρο ότι η διαδικασία που περιγράφθηκε στην παράγραφο 5.4, η οποία βασίζεται στον υπολογισμό των συντομότερων μονοπατιών σε ένα γράφημα ανάθεσης αποδίδει πολυωνυμικού χρόνου αλγόριθμο, αφού το γράφημα ανάθεσης έχει εκθετικό αριθμό από κόμβους (έναν για κάθε ανάθεση). Επομένως, θα μελετήσουμε την ανισότητα 5.7 για να προσδιορίσουμε τις αποζημιώσεις.

Υπενθυμίζουμε ότι η ωφέλεια ενός παίκτη είναι $u_i = P_i - l_i$, όπου P_i είναι οι αποζημιώσεις που καταβάλλονται στον παίκτη i. Αρχικά, για ευκολία, θα προσδιορίσουμε αρνητικές τιμές για τις αποζημιώσεις και στη συνέχεια θα δείξουμε ότι μπορούν να τροποποιηθούν έτσι ώστε οι παίκτες να έχουν μη αρνητική ωφέλεια (αν ενεργούν με φιλαλήθεια). Έστω, H^i ο αριθμός των διεργασιών που ανατίθενται στη μηχανή i στο βήμα 3. Από το πόρισμα 5.16, γνωρίζουμε ότι όλες αυτές οι διεργασίες ανατίθενται σε high μηχανές (με βάση τα δηλωμένα p_i).

Έστω $H^{-i} = \sum_{i' \neq i} H^{i'}$ και $n_{p,T}{}^{-i} = \sum_{i' \neq i} n_{p,T}{}^{i'}$. Οι αποζημιώσεις P_i στον παίκτη i καθορίζονται ως εξής:

$$P_i(p) = -L \cdot n_{p,T^*(p)}{}^{-i} - H \cdot H^{-i}(p) - (H - L)\,(n_{p,T^*(p)} - n_{p,T^L}), \qquad \text{(σχέση 5.8)}$$

όπου $(n_{p,T^*(p)} - n_{p,T^L})$, ο αριθμός των διεργασιών που από L έγιναν H μετά την περικοπή του T^* σε T^L.

Το επόμενο λήμμα επαληθεύει ότι οι αποζημιώσεις της σχέσης 5.8 δίνουν έναν φιλαλήθη μηχανισμό.

ΛΗΜΜΑ 5.24: Θεωρούμε έναν παίκτη i και τις δηλώσεις των άλλων παικτών p_{-i}. Έστω ότι ο πραγματικός τύπος του i είναι p_i^1. Τότε, με τις αποζημιώσεις που καθορίστηκαν με την (5.8), η ωφέλεια του i όταν δηλώνει τον πραγματικό του τύπο p_i^1 είναι τουλάχιστον όσο είναι η ωφελειά του όταν δηλώνει οποιονδήποτε άλλο τύπο p_i^2, δηλαδή $u(p_i^1) \geq u(p_i^2)$.

Απόδειξη: Έστω $c^1{}_i$ και $c^2{}_i$ το συνολικό κόστος του παίκτη i, που ορίζεται ως ο αρνητικός αριθμός της ωφέλειάς του, για τις εισόδους $p^1=(p_i^1, p_{-i})$ και $p^2=(p_i^2, p_{-i})$ αντίστοιχα. Αφού το p_{-i} είναι σταθερό, το παραλείπουμε από την παραπάνω έκφραση. Το πραγματικό φορτίο στον παίκτη i όταν δηλώνει τον πραγματικό του τύπο p_i^1 είναι $l_i = L \cdot n^i_{p^1,T^*(p^1)} + H \cdot H^{i}(p^1)$. Κι επομένως, το συνολικό του κόστος είναι

$$c^1{}_i = -\,u(p_i^1) = -\,P_i + l_i$$

$=L \cdot n_{p^1,T^*(p^1)}{}^{-i} + H \cdot H^{-i}(p^1) + (H - L)\,(n_{p^1,T^*(p^1)} - n_{p^1,T^L}) + L \cdot n^i{}_{p^1,T^*(p^1)} + H \cdot H^{i}(p^1)$

$=L \cdot n_{p^1,T^*(p^1)} + H \cdot (n - n_{p^1,T^*(p^1)}) + (H - L)\,(n_{p^1,T^*(p^1)} - n_{p^1,T^L}) = n \cdot H - (H - L)\, n_{p^1,T^L}$

Ενώ, το πραγματικό φορτίο στον παίκτη i όταν δηλώνει τον ψευδή τύπο p_i^2 είναι

$l_i = L \cdot (n^i{}_{p^2,T^*(p^2)} - n_H{}^{2,1} + n_L{}^{2,1} + H \cdot (H^{i} + n_H{}^{2,1} - n_L{}^{2,1})$.

Κι επομένως, το συνολικό του κόστος $c^2{}_i$ είναι

$c^2{}_i = -u(p_i^2) = -P_i + l_i$

$=L \cdot n_{p^2,T^*(p^2)}{}^{-i} + H \cdot H^{-i}(p^2) + (H - L)\,(n_{p^2,T^*(p^2)} - n_{p^2,T^L}) + L \cdot (n^i{}_{p^2,T^*(p^2)} - n_H{}^{2,1} + n_L{}^{2,1} + H \cdot (H^{i} + n_H{}^{2,1} - n_L{}^{2,1})$

$=L \cdot n_{p^2,T^*(p^2)} + (n - n_{p^2,T^*(p^2)})\, H + (H - L)\,(n_{p^2,T^*(p^2)} - n_{p^2,T^L}) + (H - L) - (H - L)\, n_L{}^{2,1}$

$=n\,H - (H - L)\, n_{p^2,T^L} + (H - L)\, n_H{}^{2,1} - (H - L)\, n_L{}^{2,1}$

Από την σχέση 5.7 έπεται ότι $c^1{}_i \le c^2{}_i \Rightarrow u(p_i^1) \ge u(p_i^2)$.

Οι αποζημιώσεις θα πρέπει να ικανοποιούν την ιδιότητα της οικειοθελούς συμμετοχής, αυτό σημαίνει ότι η ωφέλεια ενός παίκτη θα πρέπει να είναι μη αρνητική όταν αποκαλύπτει τον πραγματικό του τύπο. Οι αποζημιώσεις που δίνονται με την (5.8) δεν πληρούν την ιδιότητα, αφού ουσιαστικά χρεώνουν τον παίκτη ένα συγκεκριμένο ποσό. Γνωρίζουμε ότι αυτό το πρόβλημα λύνεται εύκολα προσθέτοντας μια αρκετά μεγάλη σταθερά στον ορισμό των αποζημιώσεων. Για παράδειγμα, στην περίπτωσή μας, έστω ότι το $\vec{H}$ συμβολίζει το διάνυσμα όλων των H διεργασιών, μπορούμε να προσθέσουμε τον όρο $n \cdot H - (H - L)\, n_{(\vec{H},p_{-i}),T^L}$ στην (5.8), που για τον παίκτη i είναι μια σταθερά.

Έτσι, οι νέες αποζημιώσεις είναι:

$Q_i(p) = n \cdot H - L \cdot n_{p,T^*(p)}^{-i} - H \cdot H^{-i}(p) - (H - L)\,(n_{p,T^*(p)} - n_{p,T^L} + n_{(\vec{H},p_{-i}),T^L})$ (σχέση 5.9)

Και το κόστος (που ορίζεται ως το αντίθετο της ωφέλειας) είναι

$c^1_i = -u(p_i^1) = -P_i + l_i =$

$-n \cdot H + L \cdot n_{p^1,T^*(p^1)}^{-i} + H \cdot H^{-i}(p^1) + (H - L)\,(n_{p^1,T^*(p^1)} - n_{p^1,T^L} + n_{(\vec{H},p_{-i}),T^L}) + L \cdot n^i_{p^1,T^*(p^1)} + H \cdot H^{i}(p^1)$

$= -n \cdot H + L \cdot n_{p^1,T^*(p^1)} + H \cdot (n - n_{p^1,T^*(p^1)}) + (H - L)\,(n_{p^1,T^*(p^1)} - n_{p^1,T^L} + n_{(\vec{H},p_{-i}),T^L})$

$= -n \cdot H + n \cdot H + (H - L)\,(n_{(\vec{H},p_{-i}),T^L} - n_{p^1,T^L})$

$= (H - L)\,(n_{(\vec{H},p_{-i}),T^L} - n_{p^1,T^L}) \leq 0$

Άρα, η ωφέλεια είναι μη αρνητική, κι επίσης εξασφαλίζεται ότι αν ένας παίκτης δε λαμβάνει καμία διεργασία τότε λαμβάνει και μηδενική αποζημίωση.

5.7. Αδυναμία Εύρεσης Αυστηρής Λύσης

Δεν υπάρχει ένας κυκλικά μονοτονικός αλγόριθμος για την περίπτωση ¨δύο-τιμών¨ L-H με λόγο προσέγγισης καλύτερο από 1,1. Έστω $H = \alpha \cdot L$ για κάποιο $2 < \alpha \leq 2.5$. Υπάρχουν δύο μηχανές I, II και εφτά διεργασίες. Θεωρούμε τα επόμενα δύο σενάρια:

Σενάριο 1: *Κάθε διεργασία έχει τον ίδιο χρόνο επεξεργασίας και στις δύο μηχανές*, οι διεργασίες 1–5, είναι L, και οι διεργασίες 6, 7 είναι H. Οποιαδήποτε βέλτιστη ανάθεση αναθέτει τις διεργασίες 1–5 στη μια μηχανή και τις διεργασίες 6, 7 στην άλλη, και έχει makespan $\mathbf{OPT_1 = 5L}$. Η δεύτερη καλύτερη ανάθεση έχει makespan τουλάχιστον $\mathbf{Second_1 = H + 3L}$.

	1	2	3	4	5	6	7
I	L	L	L	L	L	H	H
I I	L	L	L	L	L	H	H

Σενάριο 2: Αν ο αλγόριθμος επιλέγει μια βέλτιστη ανάθεση για το σενάριο 1, υποθέτουμε χωρίς βλάβη της γενικότητας ότι οι διεργασίες 6, 7 ανατίθενται στη μηχανή II. Στο σενάριο 2 η μηχανή I έχει το ίδιο διάνυσμα χρόνων επεξεργασίας. Η μηχανή II μειώνει τις διεργασίες 6, 7 σε L και αυξάνει τις διεργασίες 1–5 σε H. Μια βέλτιστη ανάθεση έχει makespan $\mathbf{OPT_2}$ = 2L + H, όπου η μηχανή II αναλαμβάνει τις διεργασίες 6, 7 και μια από τις διεργασίες 1–5. Η δεύτερη καλύτερη ανάθεση για αυτό το σενάριο έχει makespan τουλάχιστον $\text{Second}_2 = 5L$.

	1	2	3	4	5	6	7
I	L	L	L	L	L	H	H
II	H	H	H	H	H	L	L

ΘΕΩΡΗΜΑ 5.25: Κανένας ντετερμινιστικός φιλαλήθης μηχανισμός για το πρόβλημα ανάθεσης διεργασιών με ¨δύο- τιμές¨ δε μπορεί να επιτύχει λόγο προσέγγισης καλύτερο από 1,1.

Απόδειξη: Ένας κυκλικά μονοτονικός αλγόριθμος δε μπορεί να επιλέξει τη βέλτιστη ανάθεση και στα δύο σενάρια. Έστω ότι ο αλγόριθμος επιλέγει μια βέλτιστη ανάθεση για το σενάριο 1. Έστω ότι οι διεργασίες 6, 7 ανατίθενται στη μηχανή II, που το διάνυσμα με τους χρόνους επεξεργασίας τους αλλάζει στο σενάριο 2. Τώρα ο αλγόριθμος δε μπορεί να επιλέξει μια βέλτιστη λύση για το σενάριο 2, γιατί αλλιώς θα παραβιαστεί η συνθήκη της κυκλικής μονοτονικότητας για τη μηχανή II.

Έστω, p^1_{II} , p^2_{II} τα διανύσματα με τους χρόνους επεξεργασίας της μηχανής II για το σενάρια 1 και 2 αντίστοιχα, παίρνουμε $\sum_{j=1}^{7}(p^1_{II,j} - p^2_{II,j})(x^2_{II,j} - x^1_{II,j})$ =

=(L-H)(0-0)+(L-H)(0-0)+(L- H)(0-0)+ (L- H)(0-0)+ (L- H)(1-0)+ (H - L)(1-1)+ (H - L)(1-1)

= L- H < 0

Επομένως, οποιοσδήποτε φιλαλήθης μηχανισμός πρέπει να επιστρέφει ένα υπο-βέλτιστο makespan σε τουλάχιστον ένα σενάριο, κι έτσι ο λόγο προσέγγισής του είναι τουλάχιστον $\min\left\{\frac{Second_1}{OPT_1}, \frac{Second_2}{OPT_2}\right\}$, που επιτυγχάνει τη βέλτιστη τιμή 1,1 όταν α = 2,5.

ΚΕΦΑΛΑΙΟ 6. ΑΝΑΘΕΣΗ ΔΙΕΡΓΑΣΙΩΝ ΣΕ ΜΗ ΣΥΣΧΕΤΙΖΟΜΕΝΕΣ ΜΗΧΑΝΕΣ

6.1. Πιθανοτικοί Φιλαλήθεις Μηχανισμοί

Χρησιμοποιούμε τον πίνακα $t = (t_{ij})$ για να δείξουμε ένα στιγμιότυπο του προβλήματος ανάθεσης διεργασιών σε μηχανές. Θεωρούμε ότι κάθε μηχανή ελέγχεται από έναν στρατηγικό παίκτη i, ο οποίος έχει μια κρυφή πληροφορία που είναι το διάνυσμα t_i, που αποτελεί τον τύπο του. Στις μη συσχετιζόμενες μηχανές τα μέρη ενός μηχανισμού $M = (x, P)$ περιγράφονται ως εξής:

- **Αλγόριθμος Ανάθεσης**: Ο αλγόριθμος ανάθεσης x, λαμβάνει σαν είσοδο τον πίνακα με τις δηλώσεις $b = (b_1, \ldots, b_m)$ των παικτών, και δίνει σαν αποτέλεσμα μια ανάθεση που απεικονίζεται από έναν πίνακα $x = (x_{ij})$. Το x_{ij} είναι 1 εάν η διεργασία j ανατίθεται στη μηχανή i, και 0 διαφορετικά. Στην περίπτωση της κλασματικής ανάθεσης, το x_{ij} ικανοποιεί την ανισότητα $0 \leq x_{ij} \leq 1$ και δείχνει το μέρος της διεργασίας j που ανατίθεται στη μηχανή i. Κάθε διεργασία πρέπει να είναι πλήρως ανατεθειμένη, ως εκ τούτου $\sum_{j \in [n]} x_{ij} = 1, \forall i \in [m]$.

Σημειώνεται ότι κάθε x_{ij} μπορεί να αντιμετωπισθεί ως συνάρτηση του b.

- **Αλγόριθμος Κοστολόγησης**: Ο αλγόριθμος κοστολόγησης P λαμβάνει σαν είσοδο τον πίνακα με τις δηλώσεις $b = (b_1, \ldots, b_m)$ των παικτών, και δίνει σαν αποτέλεσμα ένα διάνυσμα $P = (P_1,\ldots, P_m)$, όπου το P_i αναπαριστά την αποζημίωση που θα πάρει ο παίκτης από το μηχανισμό. Κάθε P_i αντιμετωπίζεται ως συνάρτηση του b.

Ο πιθανοτικός μηχανισμός ορίζεται ως μια κατανομή μερικών ντετερμινιστικών μηχανισμών. Στον πιθανοτικό μηχανισμό, το x_{ij} είναι μια τυχαία μεταβλητή που δείχνει εάν η διεργασία j ανατίθεται στη μηχανή i. Με $P(x_{ij})$ δείχνουμε την πιθανότητα η διεργασίας j να ανατεθεί στη μηχανή i.

Στους πιθανοτικούς μηχανισμούς, υπάρχουν δύο μορφές φιλαλήθειας. Η ισχυρότερη μορφή ονομάζεται ***καθολική φιλαλήθεια*** (universally truthful), και απαιτεί ο μηχανισμός να είναι φιλαλήθης όταν ορίζονται όλες οι τυχαίες μεταβλητές, που είναι οι δηλώσεις των παικτών. Η ασθενέστερη μορφή φιλαλήθειας είναι η ***φιλαλήθεια κατά αναμενόμενη τιμή***, στην οποία απαιτείται, για κάθε παίκτη, που δηλώνει τον πραγματικό του τύπο, να μεγιστοποιείται η αναμενόμενη ωφέλεια του.

Σε έναν φιλαλήθη μηχανισμό M, μπορούμε να υποθέσουμε ότι όλοι οι παίκτες δηλώνουν τους αληθινούς τύπους τους, ως εκ τούτου b = t. Για να αξιολογηθεί η απόδοση του αλγορίθμου ανάθεσης x του μηχανισμού, εξετάζουμε το makespan. Συγκεκριμένα, δεδομένου του τύπου t του παίκτη, το makespan του μηχανισμού M απεικονίζεται με $l_M(t) = \max_{i\epsilon[m]} \sum_{j\in[n]} x_{ij} \cdot t_{ij}$. Το βέλτιστο makespan απεικονίζεται με $l_{OPT}(t) = \min_x \max_{i\epsilon[m]} \sum_{j\in[n]} x_{ij} \cdot t_{ij}$. Ένας μηχανισμός M καλείται c-προσεγγιστικός, αν για οποιοδήποτε στιγμιότυπο t, έχουμε $E[l_M(t)] \leq c \cdot l_{OPT}(t)$, όπου η αναμενόμενη τιμή προκύπτει από τις τυχαίες μεταβλητές που χρησιμοποιήθηκαν στον μηχανισμό.

ΟΡΙΣΜΟΣ 6.1 [Ανεξαρτησία από Διεργασίες]: Ένας ντετερμινιστικός μηχανισμός M είναι ανεξάρτητος των διεργασιών (task independent), εάν για οποιουσδήποτε πίνακες δηλώσεων b και b΄, και για κάθε διεργασία j, ισχύει το εξής:
ΑΝ κάθε μηχανή iϵ[m] έχει ακριβώς την ίδια δήλωση για την j και ως προς τους δυο πίνακες: $b_{ij} = b'_{ij}$,
ΤΟΤΕ η ανάθεση της διεργασίας j δεν αλλάζει, δηλαδή $x_{ij}(b) = x_{ij}(b'), \ \forall \ i\epsilon[m]$.

Στους πιθανοτικούς μηχανισμούς, υπάρχουν δύο μορφές ανεξαρτησίας διεργασιών. Η ***ασθενέστερη μορφή*** πιθανοτικών μηχανισμών ανεξάρτητων διεργασιών, είναι μια κατανομή ντετερμινιστικών μηχανισμών ανεξάρτητων διεργασιών. Η ***ισχυρότερη μορφή*** πιθανοτικών μηχανισμών ανεξάρτητων διεργασιών, όχι μόνο εξασφαλίζει ότι η ανάθεση της διεργασίας j δεν αλλάζει όταν δεν αλλάζει η στήλη j του πίνακα b, αλλά και ότι όλες οι τυχαίες μεταβλητές x_{ij} είναι ανεξάρτητες μεταξύ διαφορετικών διεργασιών. Η μελέτη μας εστιάζει στους μηχανισμούς που έχουν την ισχυρότερη μορφή ανεξαρτησίας.

Παρακάτω δίνεται το θεώρημα μονοτονικότητας που είναι το κύριο εργαλείο που χρησιμοποιείται για την εύρεση και απόδειξη κάτω φραγμάτων.

ΘΕΩΡΗΜΑ 6.2 [Θεώρημα Μονοτονικότητας]: Σε οποιοδήποτε φιλαλήθη μηχανισμό, ο αλγόριθμος ανάθεσης πρέπει να ικανοποιεί την ακόλουθη ιδιότητα μονοτονικότητας: για οποιεσδήποτε δύο δηλώσεις b και b΄, που διαφέρουν μόνο στη μηχανή i, οι αντίστοιχες αναθέσεις x(b) και x΄= x(b΄) ικανοποιούν τη σχέση [NiRo99]:

$$\sum_{j=1} (x_{ij} - x'_{ij})(b_{ij} - b'_{ij}) \leq 0$$

Η ιδιότητα μονοτονικότητας ισχύει και για τους πιθανοτικούς μηχανισμούς. Στο υπόλοιπο της εργασίας χρησιμοποιείται το ακόλουθο πόρισμα για τους φιλαλήθεις πιθανοτικούς μηχανισμούς ανεξάρτητων διεργασιών.

ΠΟΡΙΣΜΑ 6.3: Για οποιοδήποτε φιλαλήθη πιθανοτικό μηχανισμό ανεξάρτητων διεργασιών M, για οποιουσδήποτε δύο πίνακες δηλώσεων b και b΄, όπου το b΄ προκύπτει από το b, αλλάζοντας μόνο το b_{ij} με το b'_{ij}, έχουμε $(x_{ij}(b) - x_{ij}(b'))(b_{ij} - b'_{ij}) \leq 0$, όπου με x_{ij} απεικονίζεται η πιθανότητα να ανατεθεί η διεργασία j στη μηχανή i.

6.2. Ο Γενικά Πιθανοτικός Φιλαλήθης Μηχανισμός

Αρχικά θα περιγραφεί μια γενικότερη τεχνική κατασκευής μηχανισμών που ουσιαστικά μπορεί να περιγράψει και όλους τους έως τώρα γνωστούς μηχανισμούς. Στη συνέχεια και με βάση αυτό το πλαίσιο, περιγράφεται ένας βελτιωμένος φιλαλήθης μηχανισμός και μελετώνται οι περιορισμοί αυτής της προσέγγισης με την απόδειξη ενός σχεδόν αυστηρού κάτω φράγματος για όλους τους φιλαλήθεις μηχανισμούς ανεξάρτητων διεργασιών.

Έστω $f: R+ \rightarrow [0, 1]$ μια μη φθίνουσα μονότονη συνάρτηση, που ικανοποιεί την $f(0) = 0$ και $\lim_{x\to\infty} f(x) = 1$, τότε έχουμε έναν πιθανοτικό μηχανισμό M_f που εξαρτάται από την f για την ανάθεση των διεργασιών σε δύο μηχανές[LuYu08]. Η συνάρτηση f αντιμετωπίζεται ως μια αθροιστική συνάρτηση κατανομής για μια τυχαία μεταβλητή στο R^+. Ακολουθεί η περιγραφή του μηχανισμού M_F:

Είσοδος: Ο πίνακας $b = (b_{ij})$ με τις δηλώσεις των παικτών

Έξοδος: Μια τυχαία ανάθεση x και οι αποζημιώσεις $p = (p_1, p_2)$

Μηχανισμός Ανάθεσης και Κοστολόγησης:

$\mathbf{x_{1j}} \leftarrow 0, \mathbf{x_{2j}} \leftarrow 0, j = 1, 2..., n; P_1 \leftarrow 0; P_2 \leftarrow 0$

Για κάθε διεργασία $j = 1, 2..., n$ κάνε

Επέλεξε τυχαία το $s_j \in R_+$ σύμφωνα με τη συνάρτηση f

έτσι ώστε $\Pr(s_j \leq u) = f(u)$

αν $b_{1j} \leq s_j^{-1} \cdot b_{2j}$,

$\mathbf{x}_{1j} \leftarrow 1, P_1 \leftarrow P_1 + s_j^{-1} \cdot b_{2j}$;

αλλιώς

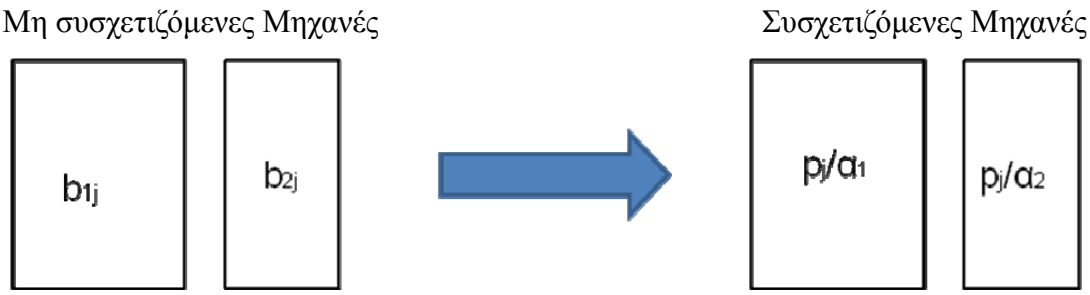
$\mathbf{x}_{2j} \leftarrow 1, P_2 \leftarrow P_2 + s_j \cdot b_{1j}$;

ΘΕΩΡΗΜΑ 6.4: Για οποιαδήποτε μη φθίνουσα μονότονη συνάρτηση $f : R+ \rightarrow [0, 1]$, όπου $f(0) = 0$ και $\lim_{x\to\infty} f(x) = 1$, ο μηχανισμός M_f είναι καθολικά φιλαλήθης (universally truthful).

Απόδειξη: Για να αποδείξουμε ότι *ο μηχανισμός* M_f είναι *καθολικά φιλαλήθης*, πρέπει να δείξουμε ότι είναι *φιλαλήθης* όταν η τυχαία ακολουθία $\{s_j\}$ είναι σταθερή.

Δεδομένου ότι η ωφέλεια ενός παίκτη είναι ίση με το άθροισμα των ωφελειών που λαμβάνονται από κάθε διεργασία και ο μηχανισμός είναι ανεξάρτητος των διεργασιών, μπορούμε να εξετάσουμε μόνο την περίπτωση μιας διεργασίας. Αν αποδείξουμε την ιδιότητα της φιλαλήθειας για την περίπτωση μιας διεργασίας j, τότε αυτή θα ισχύει και για το σύνολο των διεργασιών.

Για να αποδείξω την ιδιότητα της φιλαλήθειας, αναγάγουμε το πρόβλημα ανάθεσης της διεργασίας j σε μη συσχετιζόμενες μηχανές στο πρόβλημα ανάθεσης της διεργασίας j σε συσχετιζόμενες μηχανές, όπου με α_i συμβολίζουμε την ταχύτητα της μηχανής i και με p_j το χρόνο επεξεργασίας της διεργασίας j (β΄. σχήμα 6.1).



Σχήμα 6.1: ¨Ο Γενικός Μηχανισμός είναι Φιλαλήθης¨

Επομένως, θέτοντας όπου $a_1 = 1$, έχουμε

$$\left.\begin{aligned} b_{1j} &= \frac{p_j}{a_1} \Rightarrow \; p_j = b_{1j} a_1 \\ b_{2j} &= \frac{p_j}{a_2} \Rightarrow \; p_j = b_{2j} a_2 \end{aligned}\right\} b_{1j} a_1 = b_{2j} a_2 \Rightarrow b_{1j} = b_{2j} a_2$$

Έστω, τώρα ότι η μηχανή 1 μειώνει την ταχύτητά της α1 σε α΄1 , που σημαίνει ότι αυξάνει τον τύπο της b1j σε b΄1j. Κατά τη συνθήκη ελέγχου $b'_{1j} \le s_j^{-1} \cdot b_{2j}$ του μηχανισμού, προκύπτει μια από τις παρακάτω δύο περιπτώσεις:

- Η διεργασία δεν εκτελείται πλέον από την μηχανή 1
- Η διεργασία συνεχίζει να εκτελείται από την μηχανή 1

Καταλήγουμε στο ότι ο φόρτος εργασίας της μηχανής, μειώνεται ή παραμένει ίδιος, πάντως σίγουρα δεν αυξάνεται, και άρα φθίνει μονοτονικά, πράγμα που σημαίνει ότι ο μηχανισμός είναι υλοποιήσιμος. Σύμφωνα με το θεώρημα 3.2, οι αποζημιώσεις που κάνουν τον μηχανισμό φιλαλήθη, είναι:

$$P_i\left(b_{-i},\, b_i\right) = h_i(b_{-i}) + b_i x_i(b_{-i}, b_i) - \int_0^{b_i} x_i(b_{-i}, u)\, du$$

Οι αποζημιώσεις για την μηχανή 1 υπολογίζονται ως εξής:

1. Αν $x_1(b_1, b_2) = 1$ τότε

$$\mathrm{P}_1(\mathrm{b}_1, \mathrm{b}_2) = h_1(b_2) + b_1 x_1(b_1, b_2) - \int_0^{b_1} x_1(u, b_2)du = h_1(b_2) + b_1 - b_1 = b_2 / s$$

2. Αν $x_1(b_1, b_2) = 0$, και άρα $b_1 > b_2 / s$ τότε

$$\mathrm{P}_1(\mathrm{b}_1, \mathrm{b}_2) = h_1(b_2) + b_1 \cdot 0 - \int_0^{b_1} x_1(u, b_2)du = b_2 / s - \int_0^{b_2/s} x_1(u, b_2)du = b_2 / s - b_2 / s = 0$$

Δηλαδή αν δεν ανατεθεί καμία διεργασία στη μηχανή 1, η αποζημίωση της θα είναι μηδενική, που είναι και το ζητούμενο.

3. Στην περίπτωση όπου $j = 1, 2..., n$

Έχουμε $h_1(b_{2j}) = s_j^{-1} \cdot b_{2\mathbf{j}}$ και. $h_2(b_{1j}) = s_j \cdot b_{1\mathbf{j}}$

Κάθε γνωστός μηχανισμός μπορεί να αντιμετωπισθεί ως Mf μηχανισμός συναρτήσει της f :

$$f_1(\mathrm{x}) = \begin{cases} 1, x \geq 1, \\ 0, 0 \leq x < 1; \end{cases} \quad f_2(\mathrm{x}) = \begin{cases} 1, x \geq \frac{4}{3}, \\ \frac{1}{2}, \frac{3}{4} \leq x < \frac{4}{3}, \\ 0, 0 \leq x < \frac{3}{4}; \end{cases} \quad f_3(\mathrm{x}) = \begin{cases} 1, x \geq a, \\ r, \beta \leq x < a, \\ \frac{1}{2}, \frac{1}{\beta} \leq x < \beta, \\ 1 - r, \frac{1}{a} \leq x < \frac{1}{\beta}, \\ 0, 0 \leq x < \frac{1}{a}; \end{cases}$$

Ο M_{f_1} είναι ο Min Work Μηχανισμός, ο οποίος είναι ένας ντετερμινιστικός μηχανισμός με λόγο προσέγγισης 2[NiRo99]. Στον Min Work Μηχανισμό, η διεργασία ανατίθεται στη μηχανή με το μικρότερο χρόνο επεξεργασίας και η αποζημίωσή της ισούται με το χρόνο επεξεργασίας της δεύτερης γρηγορότερης μηχανής. Ο M_{f_2} είναι ο Biased Min Work Μηχανισμός, με λόγο προσέγγισης 1,75. Και οι δύο παραπάνω μηχανισμοί προτάθηκαν από τους Nisan και Ronen. Στη συνέχεια οι Pinyan Lu και Changyuan Yu βελτίωσαν τα αποτελέσματα των προηγούμενων μηχανισμών σχεδιάζοντας τον M_{f_3} πιθανοτικό φιλαλήθη μηχανισμό, με α = 1,4844, β = 1,1854, r = 0,7932 και λόγο προσέγγισης 1,6737. Ο αριθμός των υποπεριπτώσεων αυξάνεται πάρα πολύ σε πιο περίπλοκες συναρτήσεις.

6.3. Μέθοδος Ανάλυσης Αποδοτικότητας Μηχανισμού

ΘΕΩΡΗΜΑ 6.5 [LuYu08]: Για οποιαδήποτε μη φθίνουσα μονότονη συνάρτηση $f : R+ \to [0,\ 1],$ ο λόγος προσέγγισης του μηχανισμού Mf είναι ακριβώς $\max_{a_1, a_2 \in R_+} F(a_1, a_2)$, όπου $F : R_+ \times R_+ \to R$ ορίζεται ως εξής:

Με $r_1 = f(a_1)$ και $r_2 = f(1/a_1)$

$$F(a_1, a_2) = (1 + a_2) r_1 r_2 + r_1(1 - r_2) + (1 + a_1)(1 - r_1)(1 - r_2) + \max\{a_1, a_2\} r_2 (1 - r_1)$$

Η συνάρτηση $F(a_1, a_2)$ υπολογίζει τον λόγο προσέγγισης του μηχανισμού M_f, και έχει παραμέτρους δύο πραγματικές τιμές α (που αποτελούν φράγμα για τον τυχαίο αριθμό s_j) και τις κατανομές πιθανότητας που δίνει η f στις τιμές αυτές. Ουσιαστικά η F μετράει πόσο καλός είναι ένας μηχανισμός.

ΘΕΩΡΗΜΑ 6.6: Για $f(x)=1-\frac{1}{2^{x^{2,3}}}$, ο μηχανισμός M_f για δύο μηχανές είναι καθολικά φιλαλήθης και μπορεί να επιτύχει λόγο προσέγγισης 1,5963.

Σύμφωνα με το παρακάτω θεώρημα, επιλέγοντας $f(x)=1-\frac{1}{2^{x^{2,3}}}$, ο λόγος προσέγγισης του μηχανισμού είναι 1,5963. Η συνάρτηση που χρησιμοποιήσαμε εδώ είναι μόνο μια απεικόνιση του μηχανισμού Mf. Είναι πιθανό να υπάρχει μια καλύτερη συνάρτηση f, αν και είναι πολύ δύσκολο να βρεθεί. Επίσης, ένα ενδιαφέρον πρόβλημα είναι η εξερεύνηση της ιδιότητας της συνάρτησης f με την οποία ο μηχανισμός Mf μπορεί να επιτύχει μικρότερο λόγο προσέγγισης.

Το παρακάτω λήμμα, δίνει μια εναλλακτική περιγραφή του κανόνα ανάθεσης του μηχανισμού M_f. Δεδομένου ότι έχει ήδη αποδειχτεί ότι ο μηχανισμός είναι φιλαλήθης, η δήλωση του παίκτη μπορεί πλέον να αναπαρίσταται με t.

ΛΗΜΜΑ 6.7: Για οποιαδήποτε πίνακα με τους τύπους t των δύο μηχανών, ο μηχανισμός αναθέτει κάθε διεργασία ανεξάρτητα και για κάθε διεργασία $j = 1, 2\ldots, n$, εάν $t_{1j} = 0$, τότε αναθέτει την διεργασία πάντα στη μηχανή 1, διαφορετικά την αναθέτει στη μηχανή 1 με πιθανότητα $f(t_{2j}/t_{1j})$ και στη μηχανή 2 με πιθανότητα $1-f(t_{2j}/t_{1j})$.

Απόδειξη θεωρήματος 6.5: Έστω ότι έχουμε ένα οποιοδήποτε καθορισμένο στιγμιότυπο $t = (t_{ij})$, και ότι l_{opt} είναι το βέλτιστο makespan. Έστω O_1, O_2 είναι τα σύνολα των διεργασιών που ανατίθενται, σε μια βέλτιστη λύση, στη μηχανή 1 και τη μηχανή 2 αντίστοιχα.

Τότε έχουμε: $l_{opt} = \max\left\{\sum_{j\in O_1} t_{1j}, \sum_{k\in O_2} t_{2k}\right\}$

Το αναμενόμενο makespan του μηχανισμού M_f, απεικονίζεται με l^f . Χρησιμοποιούμε το l_i^f , i = 1,2, για να δείξουμε το χρόνο ολοκλήρωσης της μηχανής i, οπότε $l^f = \max\left\{l_1^f, l_2^f\right\}$. Έστω M μια τυχαία μεταβλητή έτσι ώστε M = 1 αν $l_1^f \geq l_2^f$, και M = 2 διαφορετικά. Επίσης, αναπαριστούμε την πιθανότητα $Pr\ (M = 1, x_{1j} = 1)$ με P_j^1 και την πιθανότητα $Pr\ (M = 2, x_{2j} = 1)$ με P_j^2, και έχουμε:

$$l^f = \sum_{j\in[m]} (t_{1j}P_j^1 + t_{2j}P_j^2)$$

$$= \sum_{j\in O_1} t_{1j}(P_j^1 + \frac{t_{2j}}{t_{1j}}P_j^2) + \sum_{k\in O_2} t_{2k}(\frac{t_{1k}}{t_{2k}}P_k^1 + P_k^2)$$

$$\leq \max_{j\in O_1}(P_j^1 + \frac{t_{2j}}{t_{1j}}P_j^2)\sum_{j\in O_1} t_{1j} + \max_{k\in O_2}(\frac{t_{1k}}{t_{2k}}P_k^1 + P_k^2)\sum_{k\in O_2} t_{2k}$$

$$\leq l_{opt}\left[\max_{j\in O_1}(P_j^1 + \frac{t_{2j}}{t_{1j}}P_j^2) + \max_{k\in O_2}(\frac{t_{1k}}{t_{2k}}P_k^1 + P_k^2)\right]$$

$$\leq l_{opt}\left[\max_{j\neq k}(P_j^1 + \frac{t_{2j}}{t_{1j}}P_j^2 + \frac{t_{1k}}{t_{2k}}P_k^1 + P_k^2)\right]$$

Επομένως, ο λόγος προσέγγισης φράσσεται από τον όρο

$$\max_{j\neq k}(P_j^1 + \frac{t_{2j}}{t_{1j}}P_j^2 + \frac{t_{1k}}{t_{2k}}P_k^1 + P_k^2)$$

Για οποιαδήποτε j, k, έστω $a_1 = \frac{t_{2j}}{t_{1j}}$, $a_2 = \frac{t_{1k}}{t_{2k}}$ και $P_{abc} = Pr(M=\alpha, x_{bj}=1, x_{ck}=1)$, α, b, c ϵ $\{1,2\}$. Τότε, για παράδειγμα, μπορούμε να επεκτείνουμε το P_j^1 σαν $P_{111} + P_{112}$.

Δηλαδή, η πιθανότητα η διεργασία j να εκτελεστεί από τη μηχανή 1, P_j^1, ισούται με την πιθανότητα η διεργασία j να εκτελεστεί από τη μηχανή 1 και η διεργασία k να εκτελεστεί από τη μηχανή 1, P_{111}, συν την πιθανότητα η διεργασία j να εκτελεστεί από τη μηχανή 1 και η διεργασία k να εκτελεστεί από τη μηχανή 2, P_{112}.

$Pr(M=1, x_{1j}=1) = Pr(M=1, x_{1j}=1, x_{1k}=1) + Pr(M=1, x_{1j}=1, x_{2k}=1)$

Έστω $r_1 = \Pr\left(M=1, x_{1j}=1\right)$ και $r_2 = \Pr\left(x_{1k}=1\right)$ τότε έχουμε

$$P_j^1 + \frac{t_{2j}}{t_{1j}} P_j^2 + \frac{t_{1k}}{t_{2k}} P_k^1 + P_k^2$$

$= (P_{111} + P_{112}) + \alpha_1(P_{221} + P_{222}) + \alpha_2(P_{111} + P_{121}) + (P_{212} + P_{222})$

$=(P_{111}+P_{112}+P_{212})$[13]$+\alpha_1 P_{221}+\alpha_1 P_{222}+(\alpha_2 P_{111}+\alpha_2 P_{121}+\alpha_2 P_{221})$[14]$-\alpha_2 P_{221}+P_{222}$

$\leq Pr(x_{1j}=1) + \alpha_2 Pr(x_{1k}=1) + (\alpha_1 - \alpha_2)P_{221} + (1 + \alpha_1)P_{222}$

$=Pr(x_{1j}=1)+\alpha_2 Pr(x_{1k}=1)+ (\alpha_1-\alpha_2)Pr(M=2,x_{2j}=1,x_{1k}=1)+(1+\alpha_1)Pr(M=2,x_{2j}=1, x_{2k}=1)$

$\leq Pr(x_{1j}=1)+ \alpha_2 Pr(x_{1k}=1) + \max\{\alpha_1-\alpha_2,0\} Pr(x_{2j}=1,x_{1k}=1)$[15]$+(1+\alpha_1)Pr(x_{2j}=1, x_{2k}=1)$

$=r_1 + \alpha_2 r_2 + \max\{\alpha_1-\alpha_2,0\}(1- r_1) r_2 +(1+\alpha_1) (1- r_1) (1- r_2)$

Με το ακόλουθο στιγμιότυπο φαίνεται ότι η ανάλυση που έγινε για τον λόγο προσέγγισης είναι ακριβής. Οι επόμενοι πίνακες απεικονίζουν τις διεργασίες και την ανάθεσή τους στις μηχανές. Υπάρχουν δύο διεργασίες A και B. Ο πρώτος πίνακας παρουσιάζει το στιγμιότυπο t, όπου $t_{1A} = 1$, $t_{1B} = \alpha_2$, $t_{2A} = \alpha_1$, $t_{2B} = 1$. Ο δεύτερος

[13] Λείπει ο όρος $P_{211,}$ δηλαδή, $Pr(x_{1j}=1) \geq (P_{111} + P_{112} + P_{212})$

[14] Λείπει ο όρος $P_{211,}$ δηλαδή, $Pr(x_{1k}=1) \geq (P_{111} + P_{121} + P_{221})$

[15] Ο όρος $Pr(M=2,x_{2j}=1,x_{1k}=1)$ είναι μικρότερος από τον όρο $Pr(x_{2j}=1,x_{1k}=1)$ και ο όρος $Pr(M=2,x_{2j}=1, x_{2k}=1)$ είναι μικρότερος από τον όρο $Pr(x_{2j}=1, x_{2k}=1)$

πίνακας δείχνει την ανάθεση αυτού του στιγμιότυπου χρησιμοποιώντας τον μηχανισμό M_f: η διεργασία Α ανατίθεται στη μηχανή 1 με πιθανότητα r1, στη μηχανή 2 με πιθανότητα 1- r_1 και η διεργασία Β ανατίθεται στη μηχανή 1 με πιθανότητα r_2, και στη μηχανή 2 με πιθανότητα 1- r_2. Εδώ $r_1 = f(a_1) \kappa\alpha\iota\ r_2 = f(1/a_1)$.

	Μηχανή 1	Μηχανή 2
Διεργασία Α	1	α_1
Διεργασία Β	α_2	1

↓

	Μηχανή 1	Μηχανή 2
Διεργασία Α	r_1	1-r_1
Διεργασία Β	r_2	1-r_2

Για να βρούμε το l_{opt}, έχουμε δύο περιπτώσεις: α) η διεργασία Α να ανατεθεί στη Μηχανή 1 και η Β στη Μηχανή 2, $l_{opt} = \max\{1,1\} = 1$ και β) η διεργασία Α να ανατεθεί στη Μηχανή 2 και η Β στη Μηχανή 1 $l_{opt} = \max\{a_1, a_2\} \le 1$. Σε αυτό το στιγμιότυπο έχουμε $l_{opt} \le 1$ και το αναμενόμενο makespan που παράγεται από τον μηχανισμό M_f είναι ακριβώς $F(a_1, a_2)$. Επομένως, ο λόγος προσέγγισης είναι τουλάχιστον $F(a_1, a_2)$.

6.4. Κάτω Φράγμα για Μηχανισμούς Ανεξάρτητων Διεργασιών

Σε αυτήν την παράγραφο θα αποδείξουμε ένα κάτω φράγμα για όλους τους φιλαλήθεις μηχανισμούς που είναι ανεξάρτητοι των διεργασιών.

ΘΕΩΡΗΜΑ 6.8: Για οποιοδήποτε φιλαλήθη μηχανισμό που είναι ανεξάρτητος των διεργασιών και για την περίπτωση των δύο μηχανών, ο λόγος προσέγγισης δεν μπορεί να είναι μικρότερος από 11/7 ($\approx 1{,}5714$).

Απόδειξη: Δεδομένου ενός οποιουδήποτε φιλαλήθη μηχανισμού M που είναι ανεξάρτητος των διεργασιών, θεωρούμε τα ακόλουθα τέσσερα στιγμιότυπα (το α είναι μια σταθερά που θα προσδιοριστεί αργότερα, και είναι $\alpha > 1$). Μπορούμε να υποθέσουμε ότι $r_1 \geq 1/2$, αλλιώς μετονομάζουμε τις μηχανές στο στιγμιότυπο 1, και αντίστοιχα τροποποιούμε τα άλλα τρία στιγμιότυπα. Σε κάθε στιγμιότυπο θα βρίσκουμε τον λόγο προσέγγισης $\frac{l_M}{l_{opt}}$.

Στιγμιότυπο 1:

	Μηχανή 1	Μηχανή 2
Διεργασία 1	1	1
Διεργασία 2	1	2

	Μηχανή 1	Μηχανή 2
Διεργασία 1	r_1	$1-r_1$
Διεργασία 2	r_2	$1-r_2$

Όλες οι δυνατές αναθέσεις είναι:

- Ανάθεση Διεργασίας 1 στη Μηχανή 1 και ανάθεση Διεργασίας 2 στη Μηχανή 1: $(1+1)r_1r_2 = \mathbf{2r_1r_2}$
- Ανάθεση Διεργασίας 1 στη Μηχανή 1 και ανάθεση Διεργασίας 2 στη Μηχανή 2: $\mathbf{2r_1(1-r_2)}$

- Ανάθεση Διεργασίας 1 στη Μηχανή 2 και ανάθεση Διεργασίας 2 στη Μηχανή 1: **$1r_2(1-r_1)$**
- Ανάθεση Διεργασίας 1 στη Μηχανή 2 και ανάθεση Διεργασίας 2 στη Μηχανή 2: **$3(1-r_1)(1-r_2)$**

Επομένως, $l_M = 2r_1r_2 + 2r_1(1-r_2) + r_2(1-r_1) + 3(1-r_1)(1-r_2)$

$= 2r_1r_2 + 2r_1 - 2r_1r_2 + r_2 - r_2r_1 + 3(1-r_1)(1-r_2)$

$= 2r_1 - r_1r_2 + r_2 + 3(1-r_1)(1-r_2) = 2r_1 + (1 - r_1)r_2 + 3(1-r_1)(1-r_2)$

Η βέλτιστη ανάθεση είναι η 3 όπου $l_{opt} = 1$.

Για αυτό το στιγμιότυπο έχουμε $\frac{l_M}{l_{opt}} = 2r_1 + (1-r_1)r_2 + 3(1-r_1)(1-r_2)$

$$\geq 1 + r_1 = A_1$$

Στιγμιότυπο 2:

	Μηχανή 1	Μηχανή 2
Διεργασία 1	1	1
Διεργασία 2	1	α

	Μηχανή 1	Μηχανή 2
Διεργασία 1	r_1	$1-r_1$
Διεργασία 2	r_3	$1-r_3$

Παρομοίως,

$l_M = 2r_1r_3 + \alpha r_1(1-r_3) + r_3(1-r_1) + (1+\alpha)(1-r_1)(1-r_3)$

$= 2r_1r_3 + \alpha r_1 - \alpha r_1r_3 + r_3 - r_1r_3 + (1-r_1+\alpha-r_1\alpha)(1-r_3)$

$= 2r_1r_3 - r_1 + \alpha - \alpha r_3 + 1$

$l_{opt} = 1$

Για αυτό το στιγμιότυπο έχουμε $\frac{l_M}{l_{opt}} = 2r_1r_3 - r_1 - \alpha r_3 + \alpha + 1 = A_2$

Στιγμιότυπο 3:

	Μηχανή 1	Μηχανή 2
Διεργασία 1	α	α^2
Διεργασία 2	1	α

	Μηχανή 1	Μηχανή 2
Διεργασία 1	r_4	$1-r_4$
Διεργασία 2	r_3	$1-r_3$

$l_M = (1+\alpha)r_4r_3 + \alpha r_4(1-r_3) + \alpha^2(1-r_4)r_3 + (\alpha^2+\alpha)(1-r_3)(1-r_4)$

$= r_4r_3 + \alpha r_4r_3 + \alpha^2 - \alpha^2r_4 + \alpha - \alpha r_3$

$l_{opt} = \alpha$

Για αυτό το στιγμιότυπο έχουμε $\frac{l_M}{l_{opt}} = (1 + 1/\alpha)r_3r_4 - r_3 - \alpha r_4 + \alpha + 1 = A_3$

Στιγμιότυπο 4:

	Μηχανή 1	Μηχανή 2
Διεργασία 1	α	α
Διεργασία 2	2α	α

	Μηχανή 1	Μηχανή 2
Διεργασία 1	r_5	$1-r_5$
Διεργασία 2	r_6	$1-r_6$

$l_M = 3\alpha r_5 r_6 + 2\alpha r_5(1-r_6) + \alpha(1-r_5)r_6 + 2\alpha(1-r_5)(1-r_6)$

$= 2\alpha r_5 r_6 - \alpha r_5 + 2\alpha$

$l_{opt} = \alpha$

Για αυτό το στιγμιότυπο έχουμε $\frac{l_M}{l_{opt}} = 2 - r_5 + 2r_5 r_6 \geq 2 - r_5 = A_4$

Στα στιγμιότυπα 3 και 4, όπου αλλάζουμε τους χρόνους επεξεργασίας της διεργασίας 2 από ¨1, α¨ σε ¨2α, α¨, θεωρούμε ότι δεν επηρεάζεται η ανάθεση της Διεργασίας 1 δεδομένου ότι ο μηχανισμός Μ είναι ανεξάρτητος των διεργασιών.

Στη συνέχεια μειώνουμε τον χρόνο επεξεργασίας της διεργασίας 1 στη Μηχανή 2 από α^2 σε α. Από το πόρισμα 3, γνωρίζουμε ότι η πιθανότητα με την οποία η Μηχανή 2 αναλαμβάνει την διεργασία 1 αυξάνεται. Αυτό σημαίνει ότι $1-r_5 \geq 1-r_4$.

Συνοψίζοντας, ο λόγος προσέγγισης του μηχανισμού M είναι τουλάχιστον $\max\{A_1, A_2, A_3, A_4\}$ με τον όρο ότι $r_1 \geq 1/2$, και $\alpha > 1$, όπου $A_1 = 1+r_1$, $A_2 = 2r_1r_3 - r_1 - \alpha r_3 + \alpha + 1$, $A_3 = (1 + 1/\alpha)r_3r_4 - r_3 - \alpha r_4 + \alpha + 1$ και $A_4 = 2 - r_4$. Επιλέγοντας $\alpha = 3/2$ αποδεικνύεται ότι $\max\{A_1, A_2, A_3, A_4\} \geq 11/7$ για οποιοδήποτε r_1, r_2, r_3, r_4.

Απόδειξη:

1. Όταν $r_1 \geq ¾$, έχουμε $A_1 \geq 7/4 > 11/7$.

2. Όταν $r_1 < ¾$, έχουμε $A_2 - A_1 = (3/2-2r_1)(1- r_3) \geq 0 \Rightarrow A_2 \geq A_1$.

   (α) όταν $r_3 < 1/2$, έχουμε $A_2 = (2r_3-1)r_1 - 3r_3/2 + 5/2 \geq (2r_3-1)\, 3/4 - 3r_3/2 + 5/2 = 7/4 > 11/7$

   (β) όταν $r_3 \geq 1/2$, έχουμε $A_2 \geq (2r_3-1)1/2 - 3r_3/2 + 5/2 = 2 - r_3/2$

3. Θεωρούμε την περίπτωση όπου $r_3 > ½$. $A_3 = (5r_3/3 - 3/2)r_4 + 5/2 - r_3$

   (α) όταν $1 \geq r_3 \geq 0.9$, r_3 μια σταθερά, A_3, A_4 δύο γραμμές. Το σημείο διασταύρωσης είναι $P\left(\frac{6r_3-3}{10r_3-3}, 1.4+\frac{1.2}{10r_3-3}\right)$. Επομένως, έχουμε $\max\{ A_3, A_4 \} \geq y_P \geq 1.4+\frac{1.2}{10-3} = 11/7$

   (β) όταν $r_3 < 0.9$, $A_3 \geq (5r_3/3 - 3/2)\cdot 1 + 5/2 - r_3 = 1 + 2r_3/3$. Επίσης, αφού έχουμε $A_2 \geq 2 - r_3/3$ όταν $r_3 \geq 1/2$. Παρομοίως, γνωρίζουμε ότι $\max\{A_3, A_2\} \geq y_Q$, όπου Q είναι το σημείο διασταύρωσης των γραμμών $y = 1 + 2r_3/3$ και $y = 2 - r_3/3$. Αφού $y_Q = 11/7$, έχουμε $\max\{A_3, A_2\} \geq 11/7$.

6.5. Ανάθεση Διεργασιών σε m Μηχανές

Ο μηχανισμός BOUNDED-SQUARE επιλύει το πρόβλημα ανάθεσης διεργασιών σε m μηχανές (δίνεται μόνο ο αλγόριθμος ανάθεσης).

Είσοδος: Ο πίνακας $b = (b_{ij})$ με τις δηλώσεις των παικτών

Έξοδος: Μια τυχαία ανάθεση $X = (X_{ij})$

Μηχανισμός Ανάθεσης:

(1) Για κάθε διεργασία $j = 1, 2..., n$ κάνε

θέσε $I_j \leftarrow \{i \epsilon [m] : b_{ij} \leq 2\min_{i \epsilon [m]} b_{ij}\}$ [16]

αν $\min_{i \epsilon [m]} b_{ij} = 0$, ανέθεσε την διεργασία j στις μηχανές του συνόλου I_j με ίση πιθανότητα;

Αλλιώς χρησιμοποίησε τον αλγόριθμο ανάθεσης SQUARE στο I_j:

Για κάθε μηχανή $i = 1, 2..., m$ κάνε:

αν $i \epsilon I_j$, $x_{ij} \leftarrow \dfrac{\dfrac{1}{(b_{ij})^2}}{\displaystyle\sum_{s \in I_j} \dfrac{1}{(b_{ij})^2}}$, αλλιώς $x_{ij} \leftarrow 0$.

(2) Μέσω διαδικασίας στρογγυλοποίησης μετέτρεψε το $(\mathbf{x}_{ij})$ σε μια τυχαία ακέραια λύση (X_{ij}) έτσι ώστε $E[X_{ij}] = \mathbf{x}_{ij}, \forall i, j$.

[16] I_j: Το σύνολο από τις πιθανές μηχανές που θα εκτελέσουν την εργασία j. Αυτές οι μηχανές θα εκτελούν την εργασία j σε χρόνο μικρότερο ή ίσο από 2 επί το μικρότερο χρόνο επεξεργασίας που έχει η συγκεκριμένη εργασία σε όλες τις μηχανές i.

Στον μηχανισμό BOUNDED-SQUARE, το $x = (x_{ij})$ αντιμετωπίζεται σαν μια κλασματική λύση του προβλήματος ανάθεσης, η οποία υιοθετείται από τον κλασματικό μηχανισμό SQUARE. Πρέπει να σημειωθεί ότι αν $x_{ij} > 0$, τότε $b_{ij} < 2\min_{i\in[m]} b_{ij} \leq 2\, l_{opt}(b)$.

ΛΗΜΜΑ 6.9: Για οποιαδήποτε μέθοδο στρογγυλοποίησης που ικανοποιεί την ισότητα $E[X_{ij}] = \mathbf{x}_{ij}, \forall i, j$ υπάρχει ένας αλγόριθμος κοστολόγησης, που χρησιμοποιεί ο μηχανισμός BOUNDED-SQUARE, ο οποίος υποστηρίζει φιλαλήθεια κατά αναμενόμενη τιμή.

Επειδή ο μηχανισμός είναι ανεξάρτητος των διεργασιών, όπως και στην περίπτωση των δύο μηχανών, για να αποδείξω την ιδιότητα της φιλαλήθειας στην ακέραια λύση, αναγάγουμε το πρόβλημα ανάθεσης της διεργασίας j σε μη συσχετιζόμενες μηχανές στο πρόβλημα ανάθεσης της διεργασίας j σε συσχετιζόμενες μηχανές. Επομένως, αποδεικνύοντας την ιδιότητα της φθίνουσας μονοτονικότητας στην κλασματική λύση μεταφέρουμε την ιδιότητα και στην ακέραια λύση μέσω της συνθήκης $E[X_{ij}] = \mathbf{x}_{ij}, \forall i, j$, που σημαίνει ότι η αναμενόμενη συνεισφορά της διεργασίας j στη μηχανή i στην ακέραια λύση ισούται με την πιθανότητα να πάρει η μηχανή αυτή τη συγκεκριμένη διεργασία j στην κλασματική λύση. Ουσιαστικά, η συνθήκη αυτή μετατρέπει τη ¨φιλαλήθεια¨ από την κλασματική ανάθεση σε ¨φιλαλήθεια κατά αναμενόμενη τιμή¨ στην ακέραια ανάθεση.

Απόδειξη: Αρχικά θα δείξουμε ότι η κλασματική ανάθεση $x = (x_{ij})$, για κάθε διεργασία j είναι μονοτονικά φθίνουσα. Αρκεί να δείξουμε ότι $x_{ij}(b) \leq x_{ij}(\tilde{b})$, όπου το $\tilde{b}$ προκύπτει από το b, απλά αντικαθιστώντας το b_{ij} με το $\tilde{b}_{ij}$, και $b_{ij} > \tilde{b}_{ij}$. Έχουμε τις παρακάτω περιπτώσεις:

Περίπτωση 1: Αν $\tilde{b}_{ij} = 0$, και κανένα άλλο $b_{sj} = 0$, $s \neq i$, τότε $x_{ij}(\tilde{b}) = 1 \geq x_{ij}(b)$.

Περίπτωση 2: Αν $\tilde{b}_{ij} = 0$, και $b_{sj} = 0$, για κάποιο $s \neq i$, τότε $x_{ij}(b) = 0 \leq x_{ij}(\tilde{b})$.

Περίπτωση 3: Αν $\tilde{b}_{ij} > 0$, $i \in I_j$, τότε $i \in \tilde{I}_j$ και $\tilde{I}_j \subset I_j$.

$$x_{ij}(b) = \frac{\frac{1}{(b_{ij})^2}}{\sum_{s \in I_j} \frac{1}{(b_{sj})^2}} \leq \frac{\frac{1}{(b_{ij})^2}}{\sum_{s \in \tilde{I}_j} \frac{1}{(b_{sj})^2}} \quad \text{αφού } \tilde{I}_j \subset I_j$$

$$= \frac{\frac{1}{(b_{ij})^2}}{\frac{1}{(b_{ij})^2} \sum_{s \in \tilde{I}_j, s \neq i} \frac{1}{(b_{sj})^2}} \leq \frac{\frac{1}{(\tilde{b}_{ij})^2}}{\frac{1}{(\tilde{b}_{ij})^2} \sum_{s \in \tilde{I}_j, s \neq i} \frac{1}{(b_{sj})^2}} = x_{ij}(\tilde{b})$$

Η τελευταία ανισότητα προκύπτει επειδή $p(u) = \frac{\frac{1}{u^2}}{\frac{1}{u^2} \sum_{s \in \tilde{I}_j, s \neq i} \frac{1}{(b_{sj})^2}}$ είναι μονοτονικά φθίνουσα όταν $u > 0$.

Περίπτωση 4: Αν $\tilde{b}_{ij} > 0$, $i \notin I_j$, τότε $x_{ij}(b) = 0 \leq x_{ij}(\tilde{b})$.

Γνωρίζουμε ότι υπάρχει ένας αλγόριθμος κοστολόγησης P_j, έτσι ώστε κάθε παίκτης να μεγιστοποιεί την ωφέλειά του από μια συγκεκριμένη διεργασία δηλώνοντας τον τύπο του με φιλαλήθεια. Αφού η διαδικασία στρογγυλοποίησης ικανοποιεί $E[X_{ij}] = \mathbf{x}_{ij}, \forall i, j$ η αναμενόμενη ωφέλεια κάθε παίκτη δε μεταβάλλεται. Χρησιμοποιώντας στον μηχανισμό αυτόν τον αλγόριθμο κοστολόγησης $\{P_j, j \epsilon [n]\}$, η ωφέλεια κάθε παίκτη είναι το άθροισμα των ωφελειών που αποκτήθηκαν από κάθε διεργασία. Επομένως, αν οι παίκτες δηλώσουν τον τύπο τους με φιλαλήθεια θα μεγιστοποιήσουν την αναμενόμενη ωφέλειά τους, που σημαίνει ότι ο μηχανισμός είναι φιλαλήθης κατά αναμενόμενη τιμή.

ΛΗΜΜΑ 6.10: *Έστω* $x = (x_{ij})$ η κλασματική λύση που προκύπτει από *τον μηχανισμό BOUNDED-SQUARE,* έχουμε $\max_{i\in[m]} \sum_{j\in[n]} x_{ij}t_{ij} \le \frac{m+1}{2} l_{opt}(t)$.

Απόδειξη: Αρχικά, θα αποδείξουμε ότι για οποιοδήποτε j, i και r, $r \neq i$, έχουμε $x_{ij}t_{ij} \le \frac{1}{2} t_{rj}$. Έστω $c_j = \min_{i\in[m]} b_{ij}$.

Περίπτωση 1: Αν $i \notin I_j$, $x_{ij}t_{ij} = 0 \le \frac{1}{2} t_{rj}$

Περίπτωση 2: Αν $i \in I_j, r \in I_j$ και $c_j > 0$, τότε

$$x_{ij}t_{ij} = \frac{\left(\frac{1}{t_{ij}}\right)^2 t_{ij}}{\sum_{s\in I_j}\left(\frac{1}{t_{sj}}\right)^2} \le \frac{\left(\frac{1}{t_{ij}}\right)^2 t_{ij}}{\left(\frac{1}{t_{rj}}\right)^2 + \left(\frac{1}{t_{ij}}\right)^2} = \frac{\frac{1}{t_{ij}}\frac{1}{t_{rj}}}{\left(\frac{1}{t_{rj}}\right)^2 + \left(\frac{1}{t_{ij}}\right)^2} t_{rj} \le \frac{1}{2} t_{rj}$$

Περίπτωση 3: Αν $i \in I_j, r \in I_j$ και $c_j = 0$, τότε $t_{ij} = t_{rj} = 0$, και $x_{ij}t_{ij} \le \frac{1}{2} t_{rj}$.

Περίπτωση 4: Αν $i \in I_j, r \notin I_j$ και $t_{ij} = c_j$, τότε $x_{ij}t_{ij} \le c_j \le \frac{1}{2} t_{rj}$.

Περίπτωση 5: Αν $i \in I_j, r \notin I_j$ και $t_{ij} > c_j$, υποθέτουμε ότι $t_{i'j} = c_j$, τότε

$$x_{ij}t_{ij} = \frac{\left(\frac{1}{t_{ij}}\right)^2 t_{ij}}{\sum_{s\in I_j}\left(\frac{1}{t_{sj}}\right)^2} \le \frac{\left(\frac{1}{t_{ij}}\right)^2 t_{ij}}{\left(\frac{1}{t_{rj}}\right)^2 + \left(\frac{1}{t_{i'j}}\right)^2} \le \frac{\left(\frac{1}{t_{ij}}\right)^2 t_{ij}}{2\left(\frac{1}{t_{ij}}\right)^2} = \frac{1}{2} t_{ij} \le \frac{1}{2} t_{rj}$$

Έστω O_r το σύνολο των διεργασιών που ανατίθενται στη μηχανή r στη βέλτιστη λύση, τότε $\sum_{j\in O_r} t_{rj} \le l_{opt}(t), \forall r.$

$$\sum_j \mathrm{x_{ij}t_{ij}} = \sum_{j\in O_i} \mathrm{x_{ij}t_{ij}} + \sum_{r\neq i}\sum_{j\in O_r} \mathrm{x_{ij}t_{ij}} \leq \sum_{j\in O_i} \mathrm{t_{ij}} + \sum_{r\neq i}\sum_{j\in O_r} \frac{1}{2}\mathrm{t_{rj}} \leq \frac{m+1}{2} l_{opt}(t)$$

ΛΗΜΜΑ 6.11: Δεδομένης μιας κλασματικής ανάθεσης x κι ενός πίνακα με χρόνους επεξεργασίας t, υπάρχει μια πιθανοτική διαδικασία στρογγυλοποίησης που παράγει μια τυχαία ακέραια ανάθεση X έτσι ώστε,

1. για οποιοδήποτε i, j $\mathrm{E[X_{ij}]} = \mathbf{x}_{ij}$
2. για οποιοδήποτε i, $\sum_{j\in[n]} X_{ij}t_{ij} < \sum_{j\in[n]} x_{ij}t_{ij} + \max_{j:x_{ij}\in(0,1)} t_{ij}$ με πιθανότητα 1

Από τον έλεγχο που κάνει ο μηχανισμός πριν δημιουργήσει το σύνολο $\mathrm{I_j}$, γνωρίζουμε ότι $\max_{j:x_{ij}\in(0,1)} t_{ij} \leq 2\, l_{opt}(\mathrm{t})$. Συνδυάζοντας τα λήμματα 6.9, 6.10 και 6.11 προκύπτει το ακόλουθο θεώρημα.

ΘΕΩΡΗΜΑ 6.12: Ο μηχανισμός BOUNDED-SQUARE είναι φιλαλήθης κατά αναμενόμενη τιμή κι έχει λόγο προσέγγισης $\frac{m+5}{2}$.

$$\sum_{j\in[n]} X_{ij}t_{ij} < \sum_{j\in[n]} x_{ij}t_{ij} + \max_{j:x_{ij}\in(0,1)} t_{ij} \leq \frac{m+1}{2} l_{opt}(t) \quad + 2\, l_{opt}(\mathrm{t}) \leq \quad \frac{m+5}{2}\, l_{opt}(\mathrm{t})$$

ΑΝΑΦΟΡΕΣ

[1] Aaron Archer, Eva Tardos. "Truthful Mechanisms for One-Parameter Agents", In Proceedings of the 42nd Symposium on Foundations of Computer Science, IEEE Computer Society, pp 482-491, 2001.

[2] Ron Lavi, Chaitanya Swamy. "Truthful Mechanism Design for Multi-Dimensional Scheduling via Cycle Monotonicity", In Proceedings of the 8th ACM conference on Electronic commerce, 2007.

[3] Pinyan Lu, Changyuan Yu. "Randomized Truthful Mechanism for Scheduling Unrelated Machines", In 4th International Workshop on Internet & Network Economics (WINE 2008), pp 231-238, 2008.

[4] Pinyan Lu, Changyuan Yu. "An Improved Randomized Truthful Mechanism for Scheduling Unrelated Machines", In Proceedings of STACS 2008, pp 527-538, 2008.

[5] Noam Nisan, Amir Ronen. "Algorithmic Mechanism Design (extended abstract)", In Proceedings of the Thirty-First Annual ACM Symposium on theory of Computing STOC '99. ACM Press, New York, pp 129-140, 1999.

[6] Noam Nisan, Tim Roughgarden, Eva Tardos, Vijay V. Vazirani. "Algorithmic Game Theory", Cambridge University Press, 2007.

[7] Martin J. Osborne, Ariel Rubinstein. "Algorithmic A Course in Game Theory", The MIT Press, 1994.

[8] L.C. Thomas. "Games, Theory and Applications", Ellis Horwood/John Wiley, Chichester, London, 1984.

[9] William Vickrey, Counterspeculation, Auctions and Competitive Sealed Tenders. Journal of Finance, 16:8-37, 1961.

[10] Rakesh V. Vohra. Paths, Cycles and Mechanism Design. Working paper, 2007.

ΣΥΝΤΟΜΟ ΒΙΟΓΡΑΦΙΚΟ

Βασικές σπουδές στο Οικονομικό Πανεπιστήμιο Αθηνών (ΑΣΟΕΕ), Πτυχίο Διοικητικής Επιστήμης και Τεχνολογίας με Ειδίκευση στη Διοίκηση Εφοδιαστικής Αλυσίδας (Logistics Management, 2005), μεταπτυχιακές σπουδές στο Πανεπιστήμιο Ιωαννίνων, Τμήμα Πληροφορικής με Ειδίκευση στη Θεωρία Επιστήμης Υπολογιστών, 2009. Πεδίο ενδιαφέροντος: Αλγοριθμικά Ζητήματα Σχεδιασμού Μηχανισμών.