ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΛΟΓΙΚΗΣ ΚΑΙ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΜΩΝ (Corelab)

Δρομολόγηση και ανάθεση μήκους κύματος σε οπτικά δίκτυα

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

Ευάγγελου Γ. Μπαμπά

Αθήνα, Οκτώβριος 2009

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΛΟΓΙΚΗΣ ΚΑΙ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΜΩΝ

Δρομολόγηση και ανάθεση μήκους κύματος σε οπτικά δίκτυα

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ

του

Ευάγγελου Γ. Μπαμπά

Συμβουλευτική Επιτροπή: Ευστάθιος Ζάχος
Τιμολέων Σελλής
Γεώργιος Κολέτσος

Εγκρίθηκε από την επταμελή εξεταστική επιτροπή:

....	....	....
Ευστάθιος Ζάχος Καθηγητής ΕΜΠ	Τιμολέων Σελλής Καθηγητής ΕΜΠ	Γεώργιος Κολέτσος Αν. Καθηγητής ΕΜΠ
....	....	....
Αριστείδης Παγουρτζής Λέκτορας ΕΜΠ	Ηλίας Κουτσουπιάς Καθηγητής ΕΚΠΑ	Ιωάννης Μήλης Αν. Καθηγητής ΟΠΑ
	....	
	Δημήτριος Φωτάκης Λέκτορας ΕΜΠ	

Αθήνα, Οκτώβριος 2009.

....
Ευάγγελος Γ. Μπαμπάς
Διδάκτωρ Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Μελετάμε μοντέλα για δρομολόγηση και ανάθεση μήκους κύματος σε οπτικά δίκτυα, με στόχο να καταδειχθούν ιδιότητες των εν λόγω μοντέλων που πρέπει να λαμβάνονται υπόψιν κατά την υλοποίηση και ανάπτυξη οπτικών δικτύων στην πράξη. Πιο συγκεκριμένα, προτείνονται προσεγγιστικοί αλγόριθμοι για τη μεγιστοποίηση του πλήθους των ικανοποιούμενων αιτήσεων σε οπτικά δίκτυα τοπολογίας δακτυλίου όπου ο αριθμός των μηκών κύματος ανά ίνα δίδεται ως μέρος της εισόδου. Οι προτεινόμενοι αλγόριθμοι, οι οποίοι έχουν όλοι φράγμενο λόγο προσέγγισης στη χειρότερη περίπτωση, συγκρίνονται και πειραματικά με ήδη γνωστούς από τη βιβλιογραφία αλγορίθμους. Από τη σύγκριση προκύπτει ότι ο αλγόριθμος με τον θεωρητικά καλύτερο λόγο προσέγγισης αποδίδει μεν καλύτερα από τους υπόλοιπους αλλά καταναλώνει υπερβολικά πολύ χρόνο. Αντίθετα, ένας από τους προτεινόμενους αλγόριθμους παράγει πολύ ικανοποιητικές λύσεις σε χρόνο που είναι αρκετές τάξεις μεγέθους μικρότερος από τον χρόνο του καλύτερου αλγορίθμου.

Επιπλέον, μελετάται μία γενίκευση του προβλήματος όπου κάθε αίτηση επικοινωνίας έχει ένα δεδομένο κέρδος, και ζητείται η μεγιστοποίηση του συνολικού κέρδους των ικανοποιούμενων αιτήσεων. Προτείνεται ένας εξαιρετικά γρήγορος, καθαρά συνδυαστικός και εύκολος στην υλοποίηση αλγόριθμος για το πρόβλημα αυτό, ο οποίος έχει χειρότερο λόγο προσέγγισης από έναν ήδη γνωστό αλγόριθμο, όμως καταφέρνει να παράγει ανταγωνιστικές λύσεις και μάλιστα σε ορισμένες περιπτώσεις καλύτερες από όλους τους άλλους αλγορίθμους που συμπεριλαμβάνονται στη μελέτη. Από την πειραματική σύγκριση προκύπτει το συμπέρασμα ότι ο προτεινόμενος αλγόριθμος αποτελεί ιδανική επιλογή όταν απαιτούνται λύσεις στο πρόβλημα σε σύντομο χρονικό διάστημα.

Μελετώνται παιγνιοθεωρητικά μοντέλα για τη δρομολόγηση και την ανάθεση μηκών κύματος σε οπτικά δίκτυα πολλαπλών ινών. Ειδικότερα, παρουσιάζεται μια πλήρης ανάλυση του κόστους της αναρχίας όταν οι παίκτες επιλέγουν εγωιστικά το μήκος κύματος ήδη δρομολογημένων αιτήσεων επικοινωνίας, χρεώνονται με βάση την μέγιστη πολλαπλότητα του μήκους κύματος που επέλεξαν κατά μήκος του μονοπατιού στο οποίο έχει δρομολογηθεί η αίτηση, και το κοινωνικό κόστος καθορίζεται από την μέγιστη πολλαπλότητα μήκους κύματος που εμφανίζεται σε ολόκληρο το δίκτυο. Αποδεικνύεται ότι το παίγνιο που ορίζεται με αυτόν τον τρόπο συγκλίνει πάντοτε σε ισορροπία Nash σε πεπερασμένο αριθμό κινήσεων, ενώ προτείνονται αλγόριθμοι για τον υπολογισμό κοινωνικά βέλτιστης ισορροπίας Nash και προσεγγιστικά βέλτιστης ισορροπίας Nash σε συγκεκριμένες τοπολογίες. Αποδεικνύεται ότι το κόστος της αναρχίας μπορεί να γίνει αυθαίρετα μεγάλο ακόμη και σε δενδρικές τοπολογίες δικτύων με μέγιστο βαθμό τρία. Όμως, στην περίπτωση του δακτυλίου και της αλυσίδας, το κόστος της αναρχίας φράσσεται από μία σταθερά αν το πλήθος των διαθέσιμων μηκών κύματος δεν είναι πολύ μεγάλο σε σχέση με το φορτίο του δικτύου, υπόθεση που καλύπτει ουσιαστικά την πλειοψηφία των περιπτώσεων που μπορεί να εμφανιστούν στην πράξη.

Προς επέκταση του προηγούμενου μοντέλου, προτείνεται ένα γενικότερο πλαίσιο μελέτης των παιγνίων εγωιστικής δρομολόγησης και ανάθεσης μηκών κύματος σε οπτικά δίκτυα πολλαπλών ινών, υπό διάφορες συναρτήσεις κόστους των παικτών και υπό διάφορες συναρτήσεις κοινωνικού κόστους. Αποδεικνύονται άνω και κάτω φράγματα για το κόστος της αναρχίας των εν λόγω παιγνίων.

Τέλος, μελετάται η πολυπλοκότητα του προβλήματος χρονικού προγραμματισμού ενός συνόλου δρομολογίων που πρέπει να εκτελούνται περιοδικά με δοσμένη συχνότητα σε ένα δίκτυο μεταφορών, έτσι ώστε να μεγιστοποιούνται οι αποστάσεις ασφαλείας μεταξύ διαδοχικών οχημάτων που χρησιμοποιούν το ίδιο τμήμα του δικτύου. Για την επίλυση αυτού του προβλήματος αποδεικνύεται και αξιοποιείται η σύνδεσή του με ένα πρόβλημα χρωματισμού μονοπατιών που έχει χρησιμοποιηθεί κατά κόρον για τη μοντελοποίηση προβλημάτων δρομολόγησης και ανάθεσης μηκών κύματος σε οπτικά δίκτυα. Έτσι, καταδεικνύεται η γενικότητα των γραφοθεωρητικών μοντέλων χρωματισμού μονοπατιών που μελετήθηκαν στη διατριβή.

Λέξεις-κλειδιά οπτικά δίκτυα, οπτικά δίκτυα πολλαπλών ινών, πολυπλεξία διαίρεσης συχνότητας, χρωματισμός μονοπατιών, χρωματισμός τόξων, πολυχρωματισμός μονοπατιών, ικανοποίηση αιτήσεων, δρομολόγηση, ανάθεση μήκους κύματος, εγωιστική δρομολόγηση, εγωιστική ανάθεση μήκους κύματος, μη συνεργατικά παίγνια, ισορροπίες Nash, κόστος αναρχίας, χρονοπρογραμματισμός τρένων, χρονοπρογραμματισμός ανεκτικός σε καθυστερήσεις, περιοδικά δρομολόγια, προσεγγιστικοί αλγόριθμοι

Abstract

We study models for routing and wavelength assignment in optical networks, aiming at showing properties of these models that must be taken into consideration when optical networks are deployed in practice. More specifically, we propose approximation algorithms for maximizing the number of satisfied requests in optical ring networks where the number of available wavelengths per fiber is given as part of the input. The proposed algorithms, which all possess a bounded approximation ratio, are also compared experimentally with other algorithms already known from the literature. From the comparison, we conclude that the algorithm with the theoretically best approximation ratio produces the best solutions but consumes too much running time. On the contrary, one of the proposed algorithms produces very satisfactory solutions with a running time several orders of magnitude faster than the time of the better algorithm.

Moreover, we study a generalization of the problem where every communication request is associated with a given profit, and we seek to maximize the total profit of satisfied requests. We propose an extremely fast, purely combinatorial, and easily implemented algorithm for this problem, which has worse approximation ratio than an already known algorithm, but manages to produce competitive solutions—in some cases, it produces better solutions than all the other algorithms included in the study. From the experimental comparison, we conclude that the proposed algorithm is a decent choice whenever we require decent solutions in limited running time.

We also study game-theoretic models for routing and wavelength assignment in multifiber optical networks. We present a full analysis of the price of anarchy when players selfishly choose the wavelength of already routed communication requests, they are charged according to the maximum fiber multiplicity incurred by their choice of wavelength, and the social cost is determined by the maximum wavelength multiplicity that appears at any edge of the network. We prove that the game thus defined always converges to a Nash equilibrium in a finite number of moves, and also propose algorithms for efficiently computing socially optimal and

approximate Nash equilibria in specific network topologies. The price of anarchy can grow unbounded even in tree networks with maximum degree three. However, in the case of chains and rings, the price of anarchy is bounded by a constant when the number of available wavelengths is not too large compared to the load of the network—an assumption which covers most cases that can appear in practice.

Extending the previous model, we propose a general framework for studying selfish routing and wavelength assignment games in multifiber optical networks, under player cost and social cost functions. We prove upper and lower bounds on the price of anarchy of these games.

Finally, we study the complexity of scheduling a set of routes that must be executed periodically in a transportation network with a given period, so that the safety distance distance between successive vehicles that use the same portion of the network is maximized. For solving this problem, we prove and utilize its connection with a path coloring problem which has been used extensively for modeling routing and wavelength assignment problems in optical networks. Thus, we show the generality of the graph-theoretic path coloring models which we studied in the thesis.

Keywords optical networks, multifiber optical networks, wavelength division multiplexing, path coloring, arc coloring, path multicoloring, request satisfaction, routing, wavelength assignment, selfish routing, selfish wavelength assignment, non-cooperative games, Nash equilibria, price of anarchy, train scheduling, delay-tolerant scheduling, periodic timetabling, approximation algorithms

στη γιαγιά μου, Λέλα

Προλεγόμενα

Η παρούσα διατριβή εκπονήθηκε στο Εργαστήριο Λογικής και Επιστήμης Υπολογισμών (Corelab) της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Εθνικού Μετσόβιου Πολυτεχνείου.[1] Το εργαστήριο διαθέτει πολυετή εμπειρία σε αλγοριθμικά θέματα δρομολόγησης και ανάθεσης μήκους κύματος σε οπτικά δίκτυα. Η συνεργασία μου με το εργαστήριο ξεκίνησε με την εκπόνηση της διπλωματικής μου εργασίας για το Δίπλωμα Ηλεκτρολόγου Μηχανικού και Μηχανικού Υπολογιστών, το φθινόπωρο του 2003. Στη συνέχεια, είχα την τιμή να γίνω δεκτός ως υποψήφιος διδάκτορας υπό την επίβλεψη του Καθηγητή ΕΜΠ κ. Στάθη Ζάχου και από το 2005 ξεκίνησα να εργάζομαι στην περιοχή που τώρα, τέσσερα χρόνια αργότερα, αποτελεί το αντικείμενο αυτής της διατριβής.

Διάρθρωση της Διατριβής

Το πρώτο κεφάλαιο χρησιμεύει ως μια μικρή εισαγωγή σε θέματα οπτικών δικτύων και θέτει τα πρώτα προβλήματα που θα μας απασχολήσουν αργότερα στη διατριβή.

Στο δεύτερο και στο τρίτο κεφάλαιο προτείνουμε προσεγγιστικούς αλγόριθμους για προβλήματα βέλτιστης δρομολόγησης και ανάθεσης μήκους κύματος σε οπτικά δίκτυα και τους συγκρίνουμε πειραματικά με άλλους αλγόριθμους που είναι ήδη γνωστοί από τη βιβλιογραφία.

Στο τρίτο και στο τέταρτο κεφάλαιο μελετάμε το κόστος αναρχίας παιγνιοθεωρητικών μοντέλων για δρομολόγηση και ανάθεση μήκους κύματος σε οπτικά δίκτυα πολλαπλών ινών.

Στο έκτο και τελευταίο κεφάλαιο ασχολούμαστε με ένα πρόβλημα από την επιστήμη μεταφορών, το οποίο μελετάμε χρησιμοποιώντας ένα μοντέλο που

[1] Η διατριβή αποτελεί υποέργο του προγράμματος ΠΕΝΕΔ 03ΕΔ/285. Το έργο συγχρηματοδοτήθηκε από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) κατά 75%, από το Ελληνικό Δημόσιο (Υπουργείο Ανάπτυξης, Γενική Γραμματεία Έρευνας και Τεχνολογίας) κατά 25% και από τον Ιδιωτικό Τομέα, στο πλαίσιο του Μέτρου 8.3 του Επιχειρησιακού Προγράμματος Ανταγωνιστικότητα—Γ' Κοινοτικό Πλαίσιο Στήριξης.

έχει χρησιμοποιηθεί κατά κόρον για την περιγραφή προβλημάτων βελτιστοποίησης σε οπτικά δίκτυα. Έτσι, καταδεικνύεται η γενικότητα των γραφοθεωρητικών μοντέλων χρωματισμού μονοπατιών με τα οποία ασχολούμαστε στη διατριβή.

Δημοσιεύσεις

Αναφέρουμε ερευνητικές εργασίες που προέκυψαν κατά τη διάρκεια της εκπόνησης της διατριβής και δημοσιεύτηκαν σε διεθνή επιστημονικά περιοδικά ή ανακοινώθηκαν σε διεθνή συνέδρια με κριτές.

1. E. Bampas, A. Pagourtzis, and K. Potika: *An experimental study of maximum profit wavelength assignment in WDM rings.* Networks, 2009 (to appear).

2. E. Bampas, L. Gąsieniec, R. Klasing, A. Kosowski, and T. Radzik: *Robustness of the rotor-router mechanism.* OPODIS, Lecture Notes in Computer Science, Springer, 2009 (to appear).

3. E. Bampas, L. Gąsieniec, N. Hanusse, D. Ilcinkas, R. Klasing, and A. Kosowski: *Euler tour lock-in problem in the rotor-router model.* DISC (Idit Keidar, ed.), Lecture Notes in Computer Science, vol. 5805, Springer, 2009, pp. 421–433.

4. E. Bampas, A. Pagourtzis, G. Pierrakos, and V. Syrganis: *Colored resource allocation games.* CTW (Sonia Cafieri, Antonio Mucherino, Giacomo Nannicini, Fabien Tarissan, and Leo Liberti, eds.), École Polytechnique and CNAM, 2009, pp. 68–72.

5. E. Bampas, A.-N. Göbel, A. Pagourtzis, and A. Tentes: *On the connection between interval size functions and path counting.* TAMC (Jianer Chen and S. Barry Cooper, eds.), Lecture Notes in Computer Science, vol. 5532, Springer, 2009, pp. 108–117.

6. E. Bampas, A. Pagourtzis, G. Pierrakos, and K. Potika: *On a noncooperative model for wavelength assignment in multifiber optical networks.* ISAAC (Seok-Hee Hong, Hiroshi Nagamochi, and Takuro Fukunaga, eds.), Lecture Notes in Computer Science, vol. 5369, Springer, 2008, pp. 159–170.

7. E. Bampas, A. Pagourtzis, and K. Potika: *Maximum profit wavelength assignment in WDM rings.* CTW (Giovanni Righini, ed.), University of Milan, 2008, pp. 35–38.

8. E. Bampas, A. Pagourtzis, and K. Potika: *Maximum request satisfaction in WDM rings: Algorithms and experiments.* PCI (Theodore S. Papatheodorou, Dimitris N. Christodoulakis, and Nikitas N. Karanikolas, eds.), Current Trends in Informatics, vol. A, New Technologies Publications, 2007, pp. 627–642.

9. E. Bampas, G. Kaouri, M. Lampis, and A. Pagourtzis: *Periodic Metro Scheduling.* ATMOS (Riko Jacob and Matthias Müller-Hannemann, eds.), Dagstuhl Seminar Proceedings, vol. 06002, Internationales Begegnungs und Forschungszentrum für Informatik (IBFI), 2006.

Ευχαριστίες

Ο επιβλέπων καθηγητής μου Στάθης Ζάχος μού άσκησε τεράστια επιρροή μέσα από τα προπτυχιακά του μαθήματα ώστε να ασχοληθώ τελικά ερευνητικά με τη Θεωρητική Πληροφορική. Είναι ένας εμπνευσμένος δάσκαλος που μεταδίδει με θέρμη την αγάπη του για τα μαθηματικά και την πληροφορική προς όλες τις κατευθύνσεις. Ο Άρης Παγουρτζής ανέλαβε μεγάλο μέρος της επίβλεψης του διδακτορικού μου και, παρά το μεγάλο φόρτο εργασίας του, ήταν πάντοτε διαθέσιμος για συζήτηση και συνεργασία.

Τα μέλη της τριμελούς συμβουλευτικής επιτροπής μου Τίμος Σελλής και Γιώργος Κολέτσος ήταν εξαιρετικά συνεργάσιμοι καθ'όλη τη διάρκεια των μεταπτυχιακών μου σπουδών. Παράλληλα, αποτέλεσε ιδιαίτερη τιμή για μένα η συμμετοχή τού Ηλία Κουτσουπιά, του Γιάννη Μήλη και του Δημήτρη Φωτάκη στην επταμελή εξεταστική επιτροπή του διδακτορικού μου.

Τα μέλη του Corelab ήταν όλα αυτά τα χρόνια ιδανικοί συνεργάτες και τους ευχαριστώ για την ατμόσφαιρα που δημιούργησαν στο εργαστήριο. Ευχαριστώ ιδιαιτέρως την Κατερίνα Ποτίκα, τον Πέτρο Ποτίκα, τον Πάνο Χείλαρη, τον Νίκο Λεονάρδο, την Γεωργία Καούρη, τον Βασίλη Ζήκα, την Βάλια Μήτσου, τον Μιχάλη Λαμπή, τον Αντώνη Αχιλλέως, τον Αντρέα Γκόμπελ, τον Άρη Τέντε, τον Γιώργο Πιερράκο και τον Θανάση Λιανέα.

Η έρευνα και, κατ'επέκταση, η ολοκλήρωση της διατριβής δεν θα ήταν δυνατή χωρίς τη βοήθεια των συνεργατών μου από το Corelab: του Άρη Παγουρτζή, της Κατερίνας Ποτίκα, της Γεωργίας Καούρη, του Μιχάλη Λαμπή, του Αντρέα Γκόμπελ, του Άρη Τέντε, του Γιώργου Πιερράκου και του Βασίλη Συργκάνη. Θέλω να αναφέρω επίσης τα μέλη και επισκέπτες της ομάδας CEPAGE του LaBRI: Ralf Klasing, David Ilcinkas, Nicolas Hanusse, Adrian Kosowski, Leszek Gąsieniec και Tomasz Radzik. Τους ευχαριστώ όλους για τις ώρες που ξοδέψαμε μαζί σκεπτόμενοι σχετικά με διάφορα προβλήματα— θεωρώ ότι ήταν από τις πιο παραγωγικές της ζωής μου. Σε αυτό το σημείο θα ήθελα να ευχαριστήσω και τον Γιάννη Εμίρη για την βοήθειά του στο να επισκεφτώ την ομάδα CEPAGE για πρακτική άσκηση την Άνοιξη του 2009.

Ευχαριστώ τα συμπαθή τετράποδα Χρυσάφη Σταμούδη, Σπύρο Μαυροκορδόπουλο και Θανάση Μακρή για τη φιλία τους και τις όμορφες στιγμές που μοιραζόμαστε όλα αυτά τα χρόνια. Ο Ισίδωρος Σιδερής, εκτός από πολύτιμος φίλος, υπήρξε και άψογος σύμβουλος για τη γραφειοκρατία του διδακτορικού και του ΠΕΝΕΔ.

Ευχαριστώ τη σύντροφό μου Βάνα για τα εκπληκτικά δύο τελευταία χρόνια, και ειδικά για την συμπαράστασή της κατά το ευαίσθητο διάστημα της συγγραφής της διατριβής. Οι μεταπτυχιακές μου σπουδές θα ήταν πολύ διαφορετικές και σίγουρα πιο δύσκολες χωρίς αυτήν.

Τέλος, οφείλω ένα τεράστιο ευχαριστώ στην οικογένειά μου που ποτέ δεν σταμάτησε να με στηρίζει με κάθε τρόπο. Η εργασία αυτή αφιερώνεται ταυτόχρονα και στους γονείς μου, Γιώργο και Καίτη.

Ευάγγελος Μπαμπάς
Αθήνα, Οκτώβριος 2009

Contents

List of Algorithms

List of Figures

List of Tables

Chapter 1

Introduction

An optical network is a communications network in which physical links between nodes of the network are implemented with optical fibers. This setup may also be referred to as an *all-optical* network in order to stress the fact that all communication is carried out in the optical domain. By contrast, in almost-all-optical networks a certain amount of electronic switching may be involved.

Optical networking is widely recognized as the technology of choice for surface communication networks. When compared to legacy copper wire, optical fibers offer huge bandwidth, low attenuation, and immunity to electromagnetic interference. On some occasions, optical fibers have been used as a simple alternative to copper wire, meaning that the optical signal used a single light frequency on the fiber and the fiber itself acted as a simple point-to-point link of high bandwidth. However, the ample bandwidth available on a single optical fiber can be exploited more efficiently: a dominating technology in contemporary all-optical networking called *Wavelength Division Multiplexing* (WDM) allows for "splitting" the fiber bandwidth into multiple independent channels (*wavelengths*), each one operating at a different light frequency. With WDM, each wavelength offers bandwidth comparable to the bandwidth that was utilized in the absence of WDM. We will refer to an all-optical network that utilizes WDM as a *WDM network*.

A communication request is described by its source node and its target node and is considered *satisfied* when the network makes it possible for a continuous data stream originating from the source to reach the target. In order to satisfy a communication request in an all-optical WDM network, one has to establish a path which connects the source to the target using the links available on the network. Following that, one also has to specify the channel that will be utilized by the data stream on each fiber of the path. This is achieved by reserving the corresponding wavelength on each fiber. The first step is referred to as "routing" and the second step is referred to as "wavelength assignment".

Technologically, it is possible to have a communication request using different wavelengths on different fibers. One way to accomplish this is by planting a piece of terminal equipment called "wavelength converter" on each node of the path where a change of wavelength is required. A second option is to convert the data stream into electronic form and then re-modulate it on a different wavelength. The first option requires expensive equipment, whereas the second option introduces unacceptable overhead because optoelectronic conversion is significantly slower than purely optical switching. For these reasons, it is common practice to enforce the constraint that a communication request must use the same wavelength on all the fibers which it traverses. Let us summarize the two constraints that limit our routing and wavelength assignment options in an all-optical WDM network:

- Each request must use a single wavelength on all the fibers that it traverses (*wavelength continuity constraint*).
- If two requests use the same fiber, then they must use different wavelengths.

The interested reader is referred to a survey by Dutta and Rouskas [29] and references therein for a broader exposition of optical network components and further options for routing in WDM networks. In the following, we will deal only with all-optical WDM networks.

1.1 Modeling an All-Optical WDM Network

A network is represented by an undirected graph $G = (V, E)$, where the set of nodes V represents the nodes of the network and the set of edges E represents the physical links of the network. The set of communication requests is represented by a set $\mathcal{R}$ of node pairs. A specific routing for a request corresponds to a simple path (i.e., a path without repetitions of nodes) on the graph, connecting its endpoints. We assume that each of the deployed fibers provides exactly the same wavelengths; if k is the number of available wavelengths per fiber, then we denote the set of available wavelengths by $W = \{a_1, \dots, a_k\}$.

Throughout this thesis, we will assume that communication requests are undirected. Undirected requests correspond to *full-duplex* communication. In this mode of communication, it is assumed that each physical link in the network is implemented with two parallel optical fibers. Each fiber is reserved for carrying data in one direction only. Whenever a request between two nodes is assigned a wavelength, this wavelength is reserved for this request on all the fibers of both parallel paths connecting the two nodes; each path is used for transferring data in one direction only. Two paths are said to *overlap* when they share a physical link of the network.

We usually identify each wavelength with a *color*. Then, the wavelength assignment problem is cast as a path coloring problem in which the following constraints must be obeyed: each path must be assigned a single color, and overlapping paths must be assigned different colors.

1.2 Optimization Problems in WDM Networks

In practice, the bandwidth available in commercially deployed WDM networks is limited to a few dozen, or at most hundred, wavelengths per fiber and the situation is not expected to change in the near future. Therefore, given a large enough network load, it will be impossible to satisfy all of the communication requests simultaneously. Accordingly, in the problems of routing and wavelength assignment in WDM networks that we will study, we will assume that the number of available wavelengths per fiber is fixed to some number k which is given as part of the input. The goal will be, then, to satisfy as many requests as possible using at most k colors. We formally define our first problem in graph-theoretic terms as follows:

Problem 1.1 (MAXIMUM ROUTING AND PATH COLORING, MAXRPC)**.**
Instance: $\langle G, \mathcal{R}, k\rangle$*, where* G *is an undirected graph,* $\mathcal{R}$ *is a set of pairs of nodes (requests), and* $k \in \mathbb{N}^+$ *is the number of available colors (wavelengths).*
Feasible solution: an assignment of paths to a subset of requests $\mathcal{R}' \subseteq \mathcal{R}$ *and a coloring of these paths with at most* k *colors so that no overlapping paths are assigned the same color.*
Goal: maximize $|\mathcal{R}'|$.

In a variation of the problem, the set of requests given as input may be already routed. Pre-routed requests arise in settings where the path on which a request will be routed is decided independently of the wavelength assignment procedure. This is the case when there are specific routing requirements, such as shortest-path routing, or when the network operator decides, for the sake of simplicity, to split the routing and wavelength assignment process into separate steps.

Problem 1.2 (MAXIMUM PATH COLORING, MAXPC)**.**
Instance: $\langle G, \mathcal{P}, k\rangle$*, where* G *is an undirected graph,* $\mathcal{P}$ *is a set of simple*

paths (pre-routed requests) defined on G, and $k \in \mathbb{N}^+$ is the number of available colors.
Feasible solution: a set of paths $\mathcal{P}' \subseteq \mathcal{P}$ that can be colored with at most k colors so that no overlapping paths are assigned the same color.
Goal: maximize $|\mathcal{P}'|$.

Note that, in general, there may be multiple paths defined on the same set of edges of a graph. We assume that each path in a given instance of the problem is distinguished by a unique identifier (ID) and thus we speak of sets instead of multisets of paths. We will not make explicit use of path ID's hereafter.

In yet another variation of the problem, each request is associated with a certain profit (or weight) and the goal is to satisfy a maximum-profit subset of the given requests. Profits may represent priorities or actual revenues associated with the communication requests. We define the MAXPROFIT-PC problem as follows:

Problem 1.3 (MAXIMUM PROFIT PATH COLORING, MAXPROFIT-PC)**.**
Instance: $\langle G, \mathcal{P}, w, k\rangle$, where G is an undirected graph, $\mathcal{P}$ is a set of simple paths defined on G, w is a profit function $w : \mathcal{P} \to \mathbb{Q}^+$, and $k \in \mathbb{N}^+$ is the number of available colors.
Feasible solution: a set of paths $\mathcal{P}' \subseteq \mathcal{P}$ that can be colored with at most k colors so that no overlapping paths are assigned the same color.
Goal: maximize $\sum_{p\in\mathcal{P}'} w(p)$.

The MAXPC, MAXRPC, and MAXPROFIT-PC problems defined above are NP-hard [75, 64] even in simple topologies such as rings and trees. An algorithm A for a maximization problem Π is a *ρ-approximation* algorithm (for $0 < \rho \leq 1$) if and only if for every input instance I of Π, A runs in time polynomial in $|I|$ (the size of the encoding of instance I) and delivers a solution with total profit at least $\rho \cdot \mathrm{OPT}$. Here, OPT denotes the profit of an optimal solution for I. Analogously, an algorithm for a minimization problem is a *ρ-approximation* algorithm for $\rho \geq 1$ if and only if it produces a solution with cost at most $\rho \cdot \mathrm{OPT}$. For a thorough introduction to NP-completeness and approximation algorithms the interested reader is referred to standard textbooks such as [38, 73].

1.3 Preliminary Definitions

We define various network topologies which will be of interest to us throughout this thesis. A *chain* is a graph that consists of a single path. A *ring* is a graph that consists of a single cycle. A *tree* is a graph in which every pair of nodes is connected by exactly one simple path. A *star* is a tree that consists of a central node (sometimes called "the hub") connected with an edge to all other nodes of the graph; these are the only edges that appear in the graph. Observe that chains and stars are special cases of trees. For a more thorough introduction to graph-theoretic concepts the reader is referred to any standard textbook on graph theory, e.g. [28].

Given a graph $G = (V, E)$ and a set of requests $\mathcal{R}$ or paths $\mathcal{P}$, we will use n for the size of set V and m for the size of set $\mathcal{R}$ or $\mathcal{P}$ (whichever is applicable). For a fixed routing of requests, we will denote by $L(e)$ the *load of an edge* $e \in E$, i.e. the number of paths that use edge e. The maximum load over all edges will be simply called *load* and denoted by L.

Chapter 2

Maximum Request Satisfaction in WDM Rings

2.1 Introduction

We follow an approach considered in several papers [65, 32, 63, 64], that of maximizing the number of requests that can be served at the same time given that the number of wavelengths is limited. Here we study the problem in rings, which is a fundamental network topology and is frequently deployed in practice (for example, in the case of SONET rings—Synchronous Optical Network rings). Moreover, the ring topology is the simplest topology where routing decisions are important, and also one of the simplest topologies for which the MAXPC and MAXRPC problems are NP-hard [75, 64].

In this chapter, we consider the two problems MAXPC and MAXRPC. Recall that in the first the routing is pre-determined and only a color assignment is sought, while in the second both a routing and a color assignment are sought. We perform an experimental evaluation of a number of algorithmic approaches for these problems in rings [10]. We first propose a new greedy heuristic for both problems which is very fast and easy to implement. We also develop improved variations of approximation algorithms that have been proposed in [75, 63, 64].

We end up with a bunch of seven algorithms for each problem. The comparative study of their performance, in terms of satisfied requests and running time, offers some interesting insights. All algorithms almost always manage to satisfy many more requests than indicated by their worst-case analysis. There are two simple algorithms that achieve satisfactory solutions very fast. The iterative algorithm usually finds largest solutions, despite the fact that it has the worst theoretical approximation ratio among the more sophisticated algorithms. One of our improved algorithms competes well with the iterative algorithm while being several times faster.

2.1.1 Related Work

MaxPC in chains is known as the "k-coloring of intervals" problem which can be solved exactly [22] in polynomial time. For MaxRPC in rings, [63] gives a $\frac{2}{3}$-approximation algorithm for the undirected problem and a $\frac{7}{11}$-approximation algorithm for the directed problem; for MaxPC in rings a $\frac{2}{3}$-approximation is described in [64]. Wan and Liu [75] present $\left(1 - \frac{1}{e}\right)$-approximation algorithms for MaxRPC in rings and for MaxPC in trees, as well as a constant approximation algorithm for MaxRPC in meshes. Their algorithms employ successive calls to algorithms that solve MaxRPC or MaxPC in instances with one available color (also known as the Maximum Edge-Disjoint Paths problem). Using the same technique, Erlebach and Jansen [33] provide a $\left(1 - \frac{1}{e}\right)$-approximation algorithm for MaxRPC in bounded-degree bidirected trees and a 0.451-approximation algorithm for general bidirected trees. The on-line version of MaxRPC has been studied in [6] where a general technique to obtain a $(\rho + 1)$-competitive algorithm for arbitrary number of wavelengths from a ρ-competitive algorithm for one wavelength is presented.

A generalization of MaxPC to multi-fiber networks has been considered for rings [66] and trees [35], where efficient constant approximation algorithms have been proposed; the problem for general topologies has been studied in [69] and [3].

2.1.2 Preliminaries

A path which is colored with some color c is called a *lonely path* if it is the only path which is colored with color c. A request is called a *lonely request* if it is routed and colored so that the corresponding path is a lonely path. Two different requests are called *compatible* (with each other) if they can be routed so that the corresponding paths are not overlapping.

Definition 2.1 (Request compatibility graph). *Let $\langle G, \mathcal{R}, k\rangle$ be an instance of* MaxRPC. *The corresponding request compatibility graph is an undirected graph $H = (\mathcal{R}, E)$, where*

$$E = \{(r, r') : r \text{ and } r' \text{ are compatible requests in } \mathcal{R}\} \quad . \tag{2.1}$$

If we remove an edge e from a ring we get a chain; we call such an edge a *separation edge*. Any separation edge induces a natural partition of the paths defined on the ring into two sets: the first set contains all those paths that actually use the separation edge, and the second set contains the rest of the paths.

Definition 2.2 (Path compatibility graph)**.** *Let $\langle G, \mathcal{P}, k \rangle$ be an instance of* MAXPC *where G is a ring, and let e be a separation edge partitioning the path set into $\mathcal{P}_e$ and $\mathcal{P}_c = \mathcal{P} \setminus \mathcal{P}_e$ where $\mathcal{P}_e$ is the set of paths using edge e. The corresponding path compatibility graph is a bipartite graph $H = (\mathcal{P}_c \cup \mathcal{P}_e, E)$, where*

$$E = \{(p, q) \in \mathcal{P}_c \times \mathcal{P}_e : p \text{ and } q \text{ do not overlap}\} \quad . \tag{2.2}$$

2.2 Algorithms for Maximum Path Coloring

In this section we focus on the case in which requests are pre-routed, that is, we study the MAXPC problem. Recall that in this case an instance actually consists of a graph G, a set of paths $\mathcal{P}$ and a number of colors k and the goal is to color as many paths as possible using the given colors, without assigning the same color to overlapping paths.

The problem can be solved exactly in $O(n + w)$ time if the input graph is a chain, using the algorithm of Carlisle and Lloyd [22]. That algorithm has the following useful property:

Property 2.3. *If $k \geq L$ the Carlisle-Lloyd algorithm colors all paths using exactly L colors. If $k < L$ the algorithm colors a maximum cardinality subset of paths of load exactly k.*

Many of the algorithms presented in this and the next chapter make use of the Carlisle-Lloyd algorithm in order to optimally color chain subinstances.

2.2.1 Shortest-First Algorithm

We first present a new, greedy algorithm for MAXPC in rings (Algorithm 1). This algorithm is easy to implement and very fast; nevertheless we will see that it is almost as competent as the more sophisticated algorithms that will follow.

We next show that this simple algorithm always achieves a solution of size at least one-third the size of an optimal solution.

Theorem 2.4. MAXPC-*SF is an $\frac{1}{3}$-approximation algorithm for the* MAXPC *problem in rings.*

Algorithm 1 MAXPC-SF

Input: an instance $\langle G, \mathcal{P}, k\rangle$ of MAXPC, where G is a ring

1: Sort paths in $\mathcal{P}$ in order of non-decreasing length.
2: **for all** paths $p \in \mathcal{P}$ **do**
3: Assign to p the smallest color that is available on all edges of p (if such a color exists, otherwise do nothing).
4: **end for**

Proof. Let $\mathcal{P}^*$ be the set of paths colored by an optimal solution and $\mathcal{P}'$ be the set of paths colored by MAXPC-SF. Let also $\mathcal{P}^*_i$ (resp. $\mathcal{P}'_i$) be the subset of $\mathcal{P}^*$ (resp. $\mathcal{P}'$) that consists of paths colored with color a_i.

Let D be any set of non-overlapping paths on the ring that remain uncolored at the end of the execution of MAXPC-SF. Fix some color a_j and consider some $p \in \mathcal{P}'_j$. If p overlaps with three or more paths from D, then at least one of them is strictly shorter than p and cannot overlap with any other path in $\mathcal{P}'_j$. This implies that MAXPC-SF would have considered this path *before* considering path p, and at that point the algorithm would have been able to color it with color a_j. This contradicts the assumption that the paths in D remain uncolored at the end of the execution of MAXPC-SF. Therefore, each path in $\mathcal{P}'_j$ overlaps with at most two paths in D. This implies that

$$|D| \leq 2 \cdot \left|\mathcal{P}'_j\right| \quad . \tag{2.3}$$

Note that Equation 2.3 holds for all j, $1 \leq j \leq k$.

Now, let $D_i = \mathcal{P}^*_i \setminus \mathcal{P}'$, that is, D_i consists of paths that are colored with a_i in the optimal solution, but were not colored by MAXPC-SF. Clearly, D_i is a set of non-overlapping paths that remain uncolored at the end of the algorithm. Consequently, from Equation 2.3, for all i, $1 \leq i \leq k$:

$$|D_i| \leq 2 \cdot \min_{1\leq j\leq k} \left|\mathcal{P}'_j\right| \leq 2 \cdot \frac{|\mathcal{P}'|}{k} \quad . \tag{2.4}$$

Let us now observe that $\mathcal{P}^* \subseteq \mathcal{P}' \cup \bigcup_{1\leq i\leq k} D_i$. This implies that

$$\left|\mathcal{P}^*\right| \leq |\mathcal{P}'| + \sum_{1\leq i\leq k} |D_i| \leq |\mathcal{P}'| + k \cdot \left(2 \cdot \frac{|\mathcal{P}'|}{k}\right) = 3 \cdot |\mathcal{P}'| \quad , \tag{2.5}$$

which completes the proof. □

Equation 2.4 implies that MAXPC-SF behaves much better on the average. For example, if some color a_j has been used fewer than $\frac{|\mathcal{P}'|}{ck}$ times, for some $c > 1$, then Equation 2.5 becomes

$$\left|\mathcal{P}^*\right| \leq \left(1 + \frac{2}{c}\right) \cdot |\mathcal{P}'| \quad ; \tag{2.6}$$

Algorithm 2 MaxPC-CombSol

Input: an instance $\langle G, \mathcal{P}, k\rangle$ of MaxPC, where G is a ring

Select as separation edge some $e \in E$ with minimum load; partition path set $\mathcal{P}$ into two path sets $\mathcal{P}_e$ and $\mathcal{P}_c$, where $\mathcal{P}_e$ contains all paths in $\mathcal{P}$ that pass through e, and $\mathcal{P}_c = \mathcal{P} \setminus \mathcal{P}_e$.
Call the Carlisle-Lloyd algorithm [22] for MaxPC in chains on input $\langle G - e, \mathcal{P}_c, k\rangle$.
Find a maximum matching M in the corresponding path compatibility graph.
Uncolor lonely paths.
while there exists an edge $e' \in M$ and free colors remain **do**
    Color the two endpoints (paths) of e' with a free color; remove e' from M.
    Uncolor lonely paths.
end while
while free colors remain **do**
    Color an uncolored path in $\mathcal{P}$ with a free color.
end while
for all colors c **do**
    Find all uncolored paths that do not overlap with any path colored with c; let $\mathcal{P}_u$ be this set.
    Find a maximum subset of non-overlapping paths of $\mathcal{P}_u$ and color them with c.
end for

that is, the solution returned is near-optimal for large c. Indeed, we will see in Section 2.4 that MaxPC-SF usually achieves quite satisfactory solutions.

A simple implementation of the algorithm has running time $O(nmk)$.

2.2.2 Combining Solutions

Algorithm MaxPC-CombSol uses two main techniques: the one colors a chain instance and the other colors pairs of non-overlapping paths. The algorithm in fact combines the two solutions so as to retain some key properties of both. This algorithm is an improved version of the algorithm presented in [64]. The main improvement is an additional last step that takes care of remaining paths that possibly lie on edges where some color is still free. Details are presented in Algorithm 2.

It has been shown in [64] that the algorithm presented there achieves an approximation guarantee of $\frac{2}{3}$. Consequently, this holds for MaxPC-

CombSol as well, since the main difference of the two algorithms is the addition of the final *for*-loop which may only augment the solution achieved in the previous steps.

Steps 12-15 of Algorithm 2 cost $O(nmk)$ time. Therefore, the time complexity of Algorithm MaxPC-CombSol is $O(nmk + m^2)$, where $O(m^2)$ is the complexity of the bipartite matching computation, using an algorithm by Ma and Spinrad [52].

Algorithm MaxPC-CombSol-all The selection of the separation edge may be crucial for the average performance of the algorithm. Therefore, we will consider a new version of the algorithm which consists of n executions of MaxPC-CombSol, each time with a different separation edge. The time complexity of this algorithm, MaxPC-CombSol-all, is $O(n(nmk + m^2))$.

2.2.3 Selecting the Best Solution

The MaxPC-BestSol algorithm is an adaptation of MaxRPC-BestSol (see Section 2.3.3). It solves each instance of the problem with two independent procedures, called Chain Step and Matching Step, and merely chooses the best solution between the solutions of these procedures. The Chain Step performs the same actions as Steps 1 and 2 of Algorithm 2. In addition, any color that remains after executing these steps is used to color a single path in $\mathcal{P}_c$. The Matching Step performs the same actions as Steps 3 and 5-6 of Algorithm 2.

It is to be noted that MaxPC-BestSol achieves the same worst-case approximation ratio as the more involved algorithm MaxPC-CombSol. The proof follows from a straightforward adaptation of the proof for the approximation ratio of MaxRPC-BestSol, which appears in [63].

Theorem 2.5. MaxPC-*BestSol is a* $\frac{2}{3}$-*approximation algorithm for the* MaxPC *problem in rings.*

The time complexity of the algorithm is determined by the bipartite matching computation which can be done in $O(m^2)$ time, using an algorithm by Ma and Spinrad [52].

Algorithm MaxPC-Chain The Chain Step of MaxPC-BestSol can be used as an algorithm on its own. Moreover, it can be shown [64] that it achieves an approximation guarantee of $\frac{1}{2}$. Its time complexity is $O(n + m)$ [64]. We will refer to this algorithm as MaxPC-Chain.

Algorithm MaxPC-BestSol-all The selection of the separation edge e may again play an important role on the average performance of the algorithm, although it does not affect the worst-case approximation ratio. Therefore, we will also evaluate a new version of the algorithm, called MaxPC-BestSol-all, which consists of n executions of MaxPC-BestSol, each time with a different separation edge. Clearly, the time complexity of MaxPC-BestSol-all is $O(nm^2)$.

2.2.4 Iterative Algorithm

Algorithm MaxPC-Iter was proposed by Wan and Liu [75] and works as follows: given a set of paths $\mathcal{P}$ and k available colors it examines colors one by one. For each color c, it computes a maximum subset $\mathcal{S}$ of non-overlapping paths. To achieve this, for each path $p \in \mathcal{P}$ it determines a maximum subset $\mathcal{S}_p$ of $\mathcal{P}$ that can be colored with the same color as p (using e.g. an algorithm for the well known Activity Selection Problem [25, p. 371]), and picks the largest such subset. It then colors paths in $\mathcal{S}$ with color c, removes $\mathcal{S}$ from $\mathcal{P}$ and proceeds with the next color.

Algorithm MaxPC-Iter achieves an approximation ratio of $1-\left(1-\frac{1}{k}\right)^k > 1-\frac{1}{e} \approx 0.632$; the ratio $1-\left(1-\frac{1}{k}\right)^k$ is slightly worse (at least for $k > 10$) than the approximation guarantee of $\frac{2}{3}$ achieved by algorithms MaxPC-BestSol, MaxPC-BestSol-all, MaxPC-CombSol, and MaxPC-CombSol-all. The time complexity of this algorithm is $O(km^2 \log m)$.

2.3 Algorithms for Maximum Routing and Path Coloring

2.3.1 Shortest-First Algorithm

We present MaxRPC-SF (see Algorithm 3), which is a heuristic analogous to the simple heuristic for MaxPC; the difference is that it also takes care of the routing by imposing shortest-path routing on all communication requests. We show that MaxRPC-SF is an $\frac{1}{5}$-approximation algorithm for MaxRPC in rings. A simple implementation of the algorithm has running time $O(nmk)$.

Theorem 2.6. MaxRPC-*SF is an* $\frac{1}{5}$*-approximation algorithm for the* MaxRPC *problem in rings.*

Proof. Let $\mathcal{R}^*$ be the set of requests satisfied by an optimal solution and $\mathcal{R}'$ be the set of requests satisfied by MaxRPC-SF. Let also $\mathcal{R}_i^*$ (resp. $\mathcal{R}_i'$) be

Algorithm 3 MAXRPC-SF

Input: an instance $\langle G, \mathcal{R}, k\rangle$ of MAXRPC, where G is a ring

Perform shortest-path routing on $\mathcal{R}$, thus obtaining a set of routed requests $\mathcal{P}$.
Sort paths in $\mathcal{P}$ in order of non-decreasing length.
for all paths $p \in \mathcal{P}$ **do**
    Assign to p the smallest color that is available on all edges of p (if such a color exists, otherwise do nothing).
end for

the subset of $\mathcal{R}^*$ (resp. $\mathcal{R}'$) that consists of requests colored with color a_i. Denote by $\breve{\mathcal{R}}_i^*$ the subset of $\mathcal{R}_i^*$ that uses longest-path routing. For any set of requests A, let $\mathrm{sp}(A)$ denote the set of paths corresponding to shortest-path routing of all requests in A.

Let $D \subseteq \mathcal{R}$ be any subset of requests that remain uncolored at the end of the execution of MAXRPC-SF, with the additional property that $\mathrm{sp}(D)$ contains non-overlapping paths. Fix some color a_j and consider some $r \in \mathcal{R}'_j$. If the shortest-path routing of r overlaps with three or more paths from $\mathrm{sp}(D)$, then at least one of them is strictly shorter than the shortest-path routing of r and cannot overlap with any other path in $\mathrm{sp}(\mathcal{R}'_j)$. This implies that MAXRPC-SF would have considered the corresponding request *before* considering request r, and at that point the algorithm would have been able to color it with color a_j. This contradicts the assumption that the requests in D remain uncolored at the end of the execution of MAXRPC-SF. Therefore, each path in $\mathrm{sp}(\mathcal{R}'_j)$ overlaps with at most two paths in $\mathrm{sp}(D)$. This implies that

$$|D| \leq 2 \cdot \left|\mathcal{R}'_j\right| \ . \tag{2.7}$$

Note that Equation 2.7 holds for all j, $1 \leq j \leq k$.

Now, let $D_i = \left(\mathcal{R}_i^* \setminus \breve{\mathcal{R}}_i^*\right) \setminus \mathcal{R}'$, that is, D_i consists of requests that are shortest-path routed and colored with a_i in the optimal solution, but were not satisfied by MAXRPC-SF. Clearly, $\mathrm{sp}(D_i)$ contains non-overlapping paths and applying Equation 2.7 we get, for all i, $1 \leq i \leq k$:

$$|D_i| \leq 2 \cdot \min_{1\leq j\leq k} \left|\mathcal{R}'_j\right| \leq 2 \cdot \frac{|\mathcal{R}'|}{k} \ . \tag{2.8}$$

Let us now observe that

$$\mathcal{R}^* = \bigcup_{1\leq i\leq k} \left(\mathcal{R}_i^* \setminus \breve{\mathcal{R}}_i^*\right) \cup \bigcup_{1\leq i\leq k} \breve{\mathcal{R}}_i^* \ . \tag{2.9}$$

By the definition of D_i, we get that $\mathcal{R}_i^* \setminus \breve{\mathcal{R}}_i^* \subseteq D_i \cup \mathcal{R}'$. Plugging this into Equation 2.9, we get that

$$\mathcal{R}^* \subseteq \bigcup_{1\le i\le k} (D_i \cup \mathcal{R}') \cup \bigcup_{1\le i\le k} \breve{\mathcal{R}}_i^* = \mathcal{R}' \cup \bigcup_{1\le i\le k} D_i \cup \bigcup_{1\le i\le k} \breve{\mathcal{R}}_i^* \ . \tag{2.10}$$

Therefore,

$$\left|\mathcal{R}^*\right| \le |\mathcal{R}'| + \sum_{1\le i\le k} |D_i| + \sum_{1\le i\le k} \left|\breve{\mathcal{R}}_i^*\right| \ . \tag{2.11}$$

Finally, note that if $\left|\mathcal{R}_i^*\right| \ge 3$, then at most one request in $\mathcal{R}_i^*$ can be longest-path routed. So, in any case,

$$\left|\breve{\mathcal{R}}_i^*\right| \le 2 \ . \tag{2.12}$$

Plugging Equations 2.8 and 2.12 into Equation 2.11, we have that

$$\left|\mathcal{R}^*\right| \le |\mathcal{R}'| + k \cdot \left(2 \cdot \frac{|\mathcal{R}'|}{k}\right) + 2k = 3 \cdot |\mathcal{R}'| + 2k \ . \tag{2.13}$$

This completes the proof because either $k \le |\mathcal{R}'|$, whence $\left|\mathcal{R}^*\right| \le 5 \cdot |\mathcal{R}'|$, or there are unused colors at the end of the execution of MaxRPC-SF, which implies that the solution returned by the algorithm is optimal. □

2.3.2 Combining Solutions

Our second algorithm for MaxRPC (MaxRPC-CombSol, see Algorithm 4) is the analogue of MaxPC-CombSol for the MaxRPC problem.

It can be shown that MaxRPC-CombSol returns a solution which is at least as large as the solution returned by a $\frac{2}{3}$-approximation algorithm for MaxRPC in rings that was presented in [63] (we will also implement that algorithm under the name MaxRPC-BestSol; see Section 2.3.3). Therefore, MaxRPC-CombSol is a $\frac{2}{3}$-approximation algorithm.

Steps 5-15 of Algorithm 4 cost $O(nmk)$ time. Therefore, the time complexity of Algorithm MaxRPC-CombSol is $O(nmk + m^3)$, where $O(m^3)$ is the time complexity of the maximum matching computation of Step 3.

Algorithm MaxRPC-CombSol-all As before, we will also consider an algorithm consisting of n calls to MaxRPC-CombSol, each with a different separation edge; we call this algorithm MaxRPC-CombSol-all. The time complexity of MaxRPC-CombSol-all is $O(n(nmk + m^3))$.

Algorithm 4 MAXRPC-CombSol

Input: an instance $\langle G, \mathcal{R}, k \rangle$ of MAXRPC, where G is a ring

Select as separation edge some $e \in E$ with minimum load with respect to shortest-path routing; route requests so that paths avoid e; let $\mathcal{P}_c$ denote the resulting set of paths.

Call the Carlisle-Lloyd algorithm [22] for MAXPC in chains on input $\langle G - e, \mathcal{P}_c, k \rangle$.

Find a maximum matching M in the corresponding request compatibility graph.

Uncolor lonely requests.

while there exists an edge $e' \in M$ with at least one endpoint uncolored and free colors remain **do**

    Color the two endpoints (requests) of e' with a free color; route the requests accordingly; remove e' from M.

    Uncolor lonely requests.

end while

while free colors remain **do**

    Select an uncolored request in $\mathcal{R}$, route it using the shortest path and color it with a free color.

end while

for all colors c **do**

    Find all uncolored requests that can be routed without overlapping with any path colored with c; route them accordingly; let $\mathcal{P}_u$ be the resulting set of paths.

    Find a maximum subset of non-overlapping paths of $\mathcal{P}_u$ and color them with c.

end for

2.3.3 Selecting the Best Solution

Algorithm MaxRPC-BestSol was presented in [63]. The Chain Step is the same as Steps 1 and 2 of Algorithm 4. The Matching Step is the same as Steps 3 and 5-6 of Algortihm 4. As in the case of MaxPC-BestSol, we independently call the Chain Step and the Matching Step and choose the best between the two solutions. As shown in [63], MaxPC-BestSol is a $\frac{2}{3}$-approximation algorithm. The time complexity of MaxRPC-BestSol is determined by the maximum matching computation which can be done in $O(m^3)$ time.

Algorithms MaxRPC-BestSol-all and MaxRPC-Chain In the same manner as in the case of MaxPC-BestSol, we consider the variation of MaxRPC-BestSol consisting of n calls to MaxRPC-BestSol, each time with a different separation edge; we call this algorithm MaxRPC-BestSol-all. We also consider the Chain Step of MaxRPC-BestSol as a separate algorithm, called MaxRPC-Chain. It was shown in [63] that MaxRPC-Chain is a $\frac{1}{2}$-approximation algorithm. The time complexity of MaxRPC-BestSol-all is $O(nm^3)$, and the time complexity of MaxRPC-Chain is $O(n + m)$.

2.3.4 Iterative Algorithm

Wan and Liu [75] have also proposed an algorithm for MaxRPC in rings, which we will refer to as MaxRPC-Iter. The algorithm works similarly to MaxPC-Iter, except that for each color c and for each request r it examines two paths: the first corresponds to the clockwise routing of r, while the second corresponds to the counter-clockwise routing of r. In each case, every other request is routed so as to avoid overlapping with the path assigned to r, or it is ignored if no such routing exists. After considering all routings obtained as above, the one that routes a maximum subset $\mathcal{S}$ of requests is chosen and the corresponding paths are colored with the current color c. The requests in $\mathcal{S}$ are then removed from the input and the algorithm proceeds with the next color. Similarly to MaxPC-Iter, the iterative algorithm for MaxRPC in rings achieves an approximation ratio of $1 - \left(1 - \frac{1}{k}\right)^k > 1 - \frac{1}{e} \approx 0.632$ and its time complexity is $O(km^2 \log m)$.

2.4 Numerical Results

We implemented all algorithms in C++, making use of the LEDA™ class library of efficient data types and algorithms. All source files were compiled with the Borland™ C++ 5.5 for Win32 compiler, set to generate fastest possible code. We relied on LEDA routines and classes for graph, array, list and priority queue operations including sorting and finding matchings in general graphs. The experiments were run on a Pentium™ 4 clocked at 3.2GHz with 512MB of memory.

For each combination of number of nodes (n), number of paths/requests (m) and number of available wavelengths (k), we randomly generated two sets of 60 instances each. For the first set, we assumed uniform distribution of the endpoints of the paths/requests over the nodes of the ring. For the second set, we assumed normal distribution with standard deviation $\sigma \approx \frac{2n}{15}$. We executed each algorithm on these sets of instances and measured the average execution time and the average number of satisfied paths/requests. Furthermore, for each of these values we calculated a 95 percent confidence interval which is shown on the plots. In the plots where we show the number of satisfied paths/requests, we include a computed upper bound for the sake of comparison.

Note that execution times were measured using the *timer* class of the LEDA package, which does not provide for measuring exact processor time. However, we ran the experiments on a dedicated machine in order to keep background processes at a minimum.

Computing an upper bound on OPT In order to obtain an estimation of the performance of our algorithms we propose an efficient way to compute an upper bound on the value of an optimal solution. We denote by $|p|$ the *length* of path p, i.e. the number of edges that path p uses. If requests are given instead of paths (MAXRPC problem) we consider for each request $r = (i, j)$ the shortest path p between nodes i and j. We index all paths in non-decreasing order of their length. It can be easily proven that the following lemma holds:

Lemma 2.7. *Let B be the smallest number such that $\sum_{i=1}^{B+1} |p_i| > nk$. Then B is an upper bound on the number of paths (requests) that can be satisfied with k colors.*

2.4.1 Discussion

A first observation is that all algorithms perform considerably better than their theoretical guarantee. Indeed, we have included a curve showing the computed upper bound (UB) in our figures and it turns out that all algorithms manage to satisfy a good fraction of an optimal solution, very often better than the theoretically predicted. Taking also into account that the upper bound used may be overestimated it is possible that the actual performance of the algorithms is even better.

The experimental comparison of the algorithms shows that each algorithm for MaxPC has similar behaviour to the corresponding algorithm for MaxRPC, on instances of similar size. Note, however, that the latter is slightly slower (since routing is involved) but usually achieves a higher number of satisfied requests due to the freedom of choosing a more adequate routing for each request.

Clearly, the best algorithms in terms of number of satisfied requests are Max(R)PC-Iter and Max(R)PC-CombSol-all (see Figures 2.1 and 2.2). However, it is clear from Figure 2.3 that Max(R)PC-CombSol-all has the worst time complexity and Max(R)PC-Iter, while faster than Max(R)PC-CombSol-all, is still quite slow when compared to Max(R)PC-SF, Max(R)PC-Chain, Max(R)PC-BestSol, and Max(R)PC-CombSol. Among the latter, Max(R)PC-CombSol appears extremely competitive. In fact, in terms of performance per time unit spent Max(R)PC-CombSol is clearly the best of all algorithms. Max(R)PC-Chain seems to be even better with respect to performance/time ratio, providing solutions that can be considered satisfactory very fast; however its performance decreases linearly as the number of wavelengths increases (see Figure 2.4). In contrast, algorithm Max(R)PC-SF, which is also extremely fast, remains relatively competitive even for large number of wavelengths. Finally, Max(R)PC-BestSol has practically the same performance as the much faster Max(R)PC-Chain and Max(R)PC-BestSol-all has poor performance while being the most time-consuming algorithm.

It is rather surprising that for large values of k Max(R)PC-Iter exhibits a clear superiority although it has the theoretically worst approximation ratio among all the algorithms (with the exception of Max(R)PC-Chain and Max(R)PC-SF). Figure 2.4 shows that the superiority of Max(R)PC-Iter increases as k increases, but its time complexity depends linearly on k while all other algorithms are practically independent from k (probably with the exception of Max(R)PC-CombSol-all). Besides, Max(R)PC-Iter exhibits a super-quadratic dependence on the number of requests m, as shown in Figures 2.3 and 2.5. A similar dependence on m alse characterizes Max(R)PC-CombSol-all and Max(R)PC-BestSol-all, while all other algorithms seem to have a sub-quadratic dependence on m.

Finally, Figure 2.5 shows that the time complexity of Max(R)PC-Iter has a super-quadratic dependence on m (the number of requests), the time complexity of Max(R)PC-Chain and Max(R)PC-SF is almost linear on m and the time complexities of the remaining algorithms are quadratic on m. These differences may become crucial for very large numbers of requests.

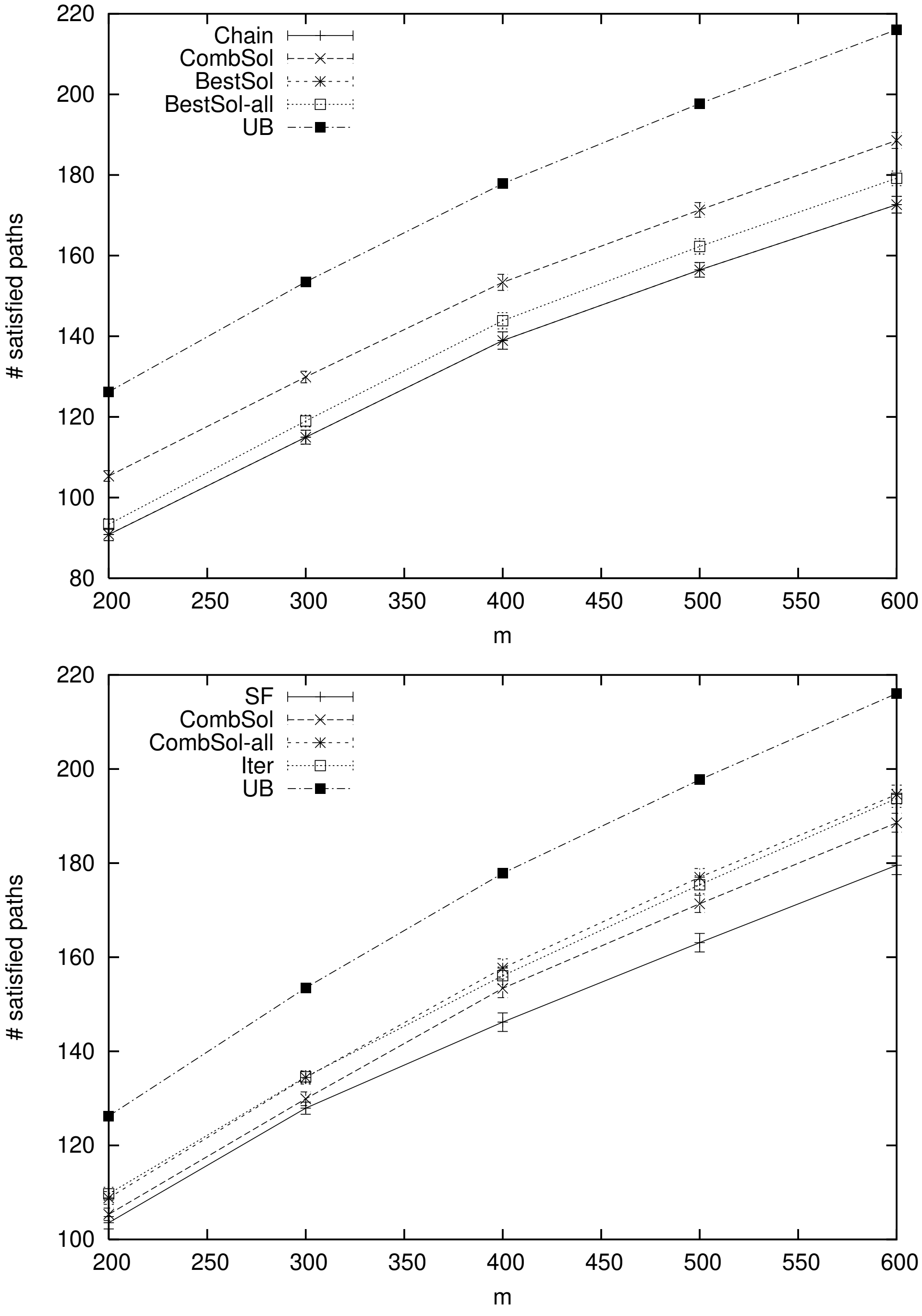

Figure 2.1: Performance of algorithms for MAXPC in terms of the number of satisfied paths: $n = 100$, m ranges from 200 to 600, $k = 40$, uniform distribution. *Top*: MAXPC-Chain, MAXPC-CombSol, MAXPC-BestSol, and MAXPC-BestSol-all. *Bottom*: MAXPC-SF, MAXPC-CombSol, MAXPC-CombSol-all, and MAXPC-Iter.

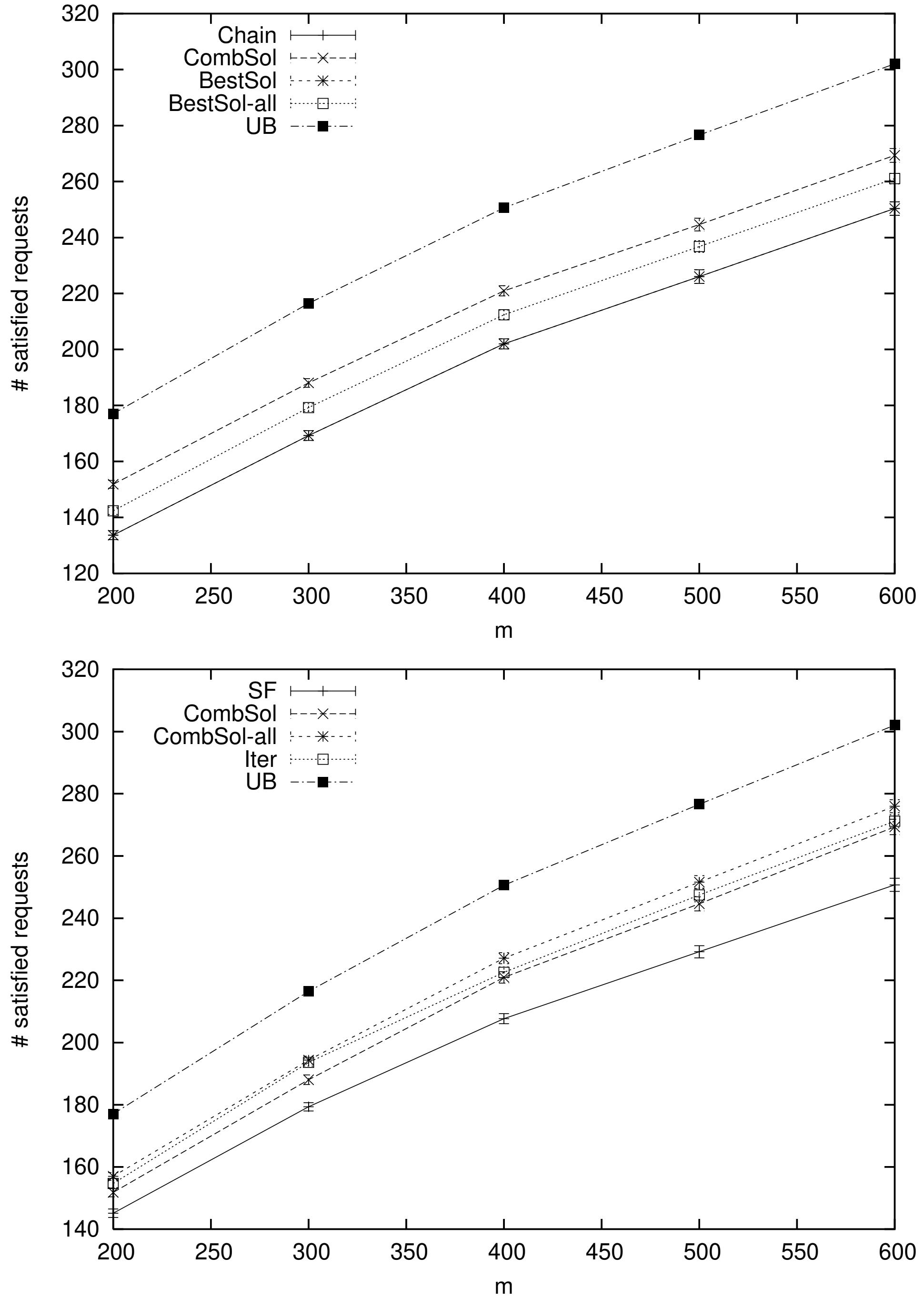

Figure 2.2: Performance of algorithms for MAXRPC in terms of the number of satisfied requests: $n = 100$, m ranges from 200 to 600, $k = 40$, uniform distribution. *Top*: MAXRPC-Chain, MAXRPC-CombSol, MAXRPC-BestSol, and MAXRPC-BestSol-all. *Bottom*: MAXRPC-SF, MAXRPC-CombSol, MAXRPC-CombSol-all, and MAXRPC-Iter.

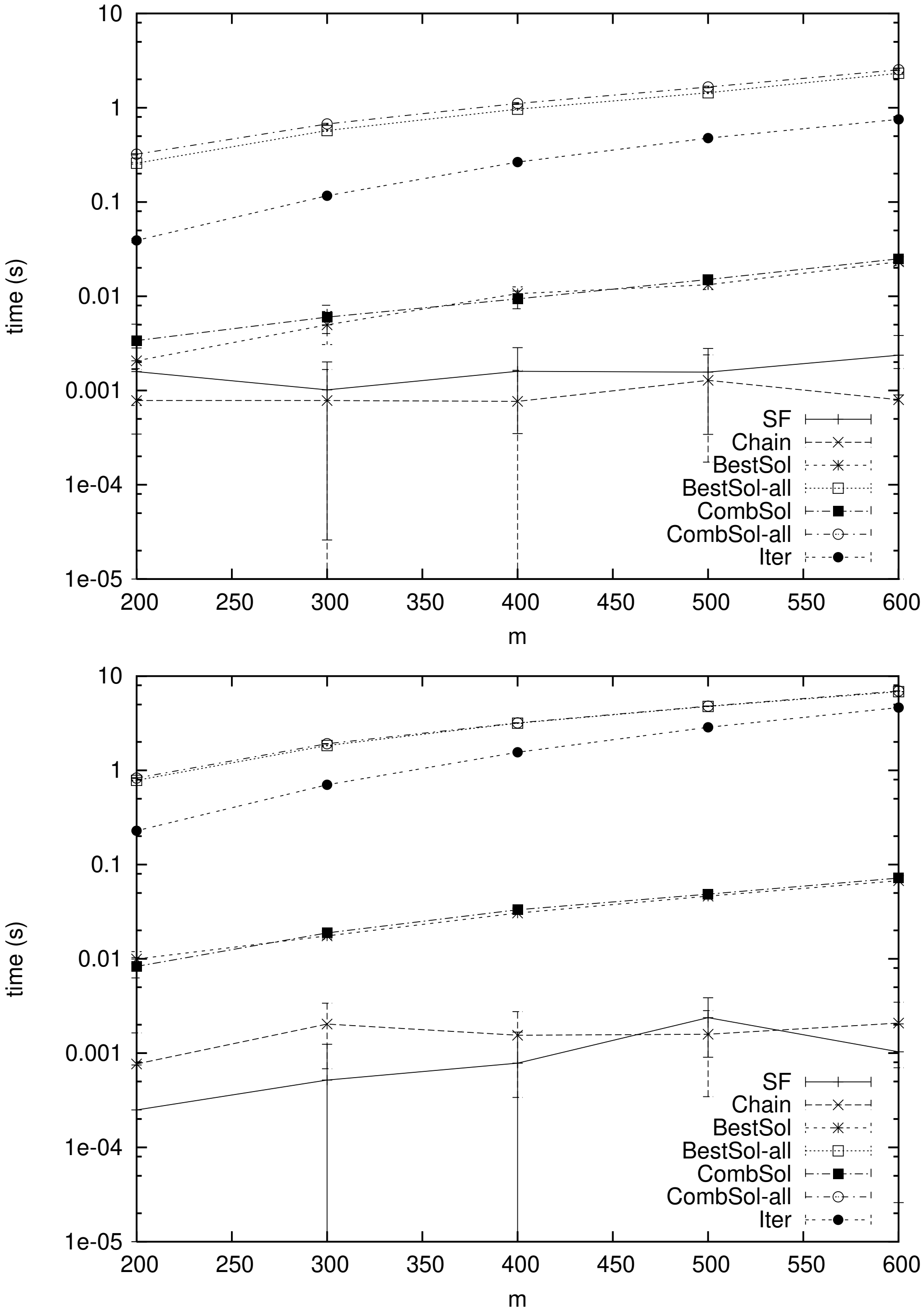

Figure 2.3: Time performance of algorithms for MaxPC (*top*) and MaxRPC (*bottom*): $n = 100$, m ranges from 200 to 600, $k = 40$, uniform distribution.

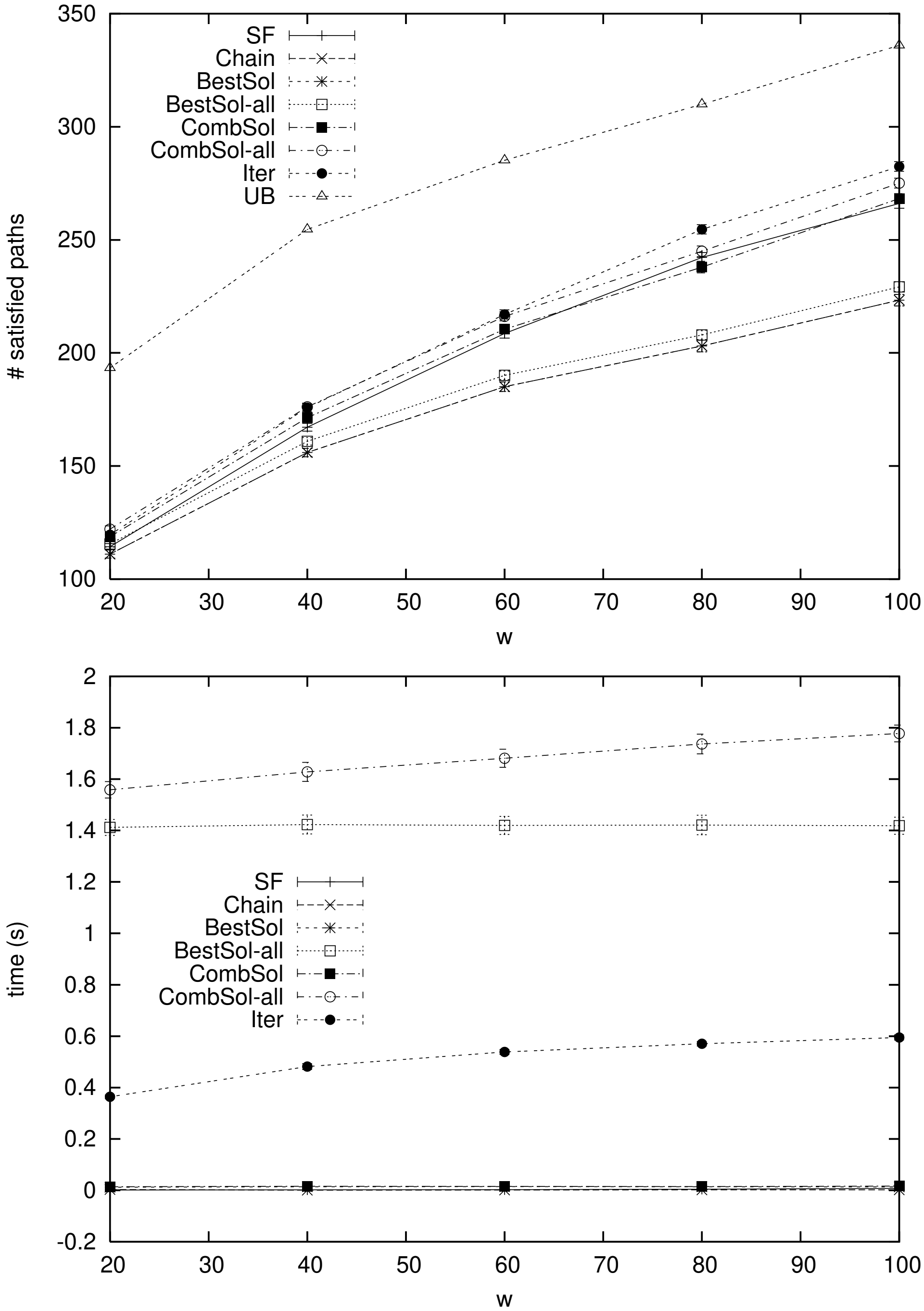

Figure 2.4: Performance of algorithms for MaxPC in terms of the number of satisfied paths (*top*) and time (*bottom*): $n = 100$, $m = 500$, k ranges from 20 to 100, Gaussian distribution.

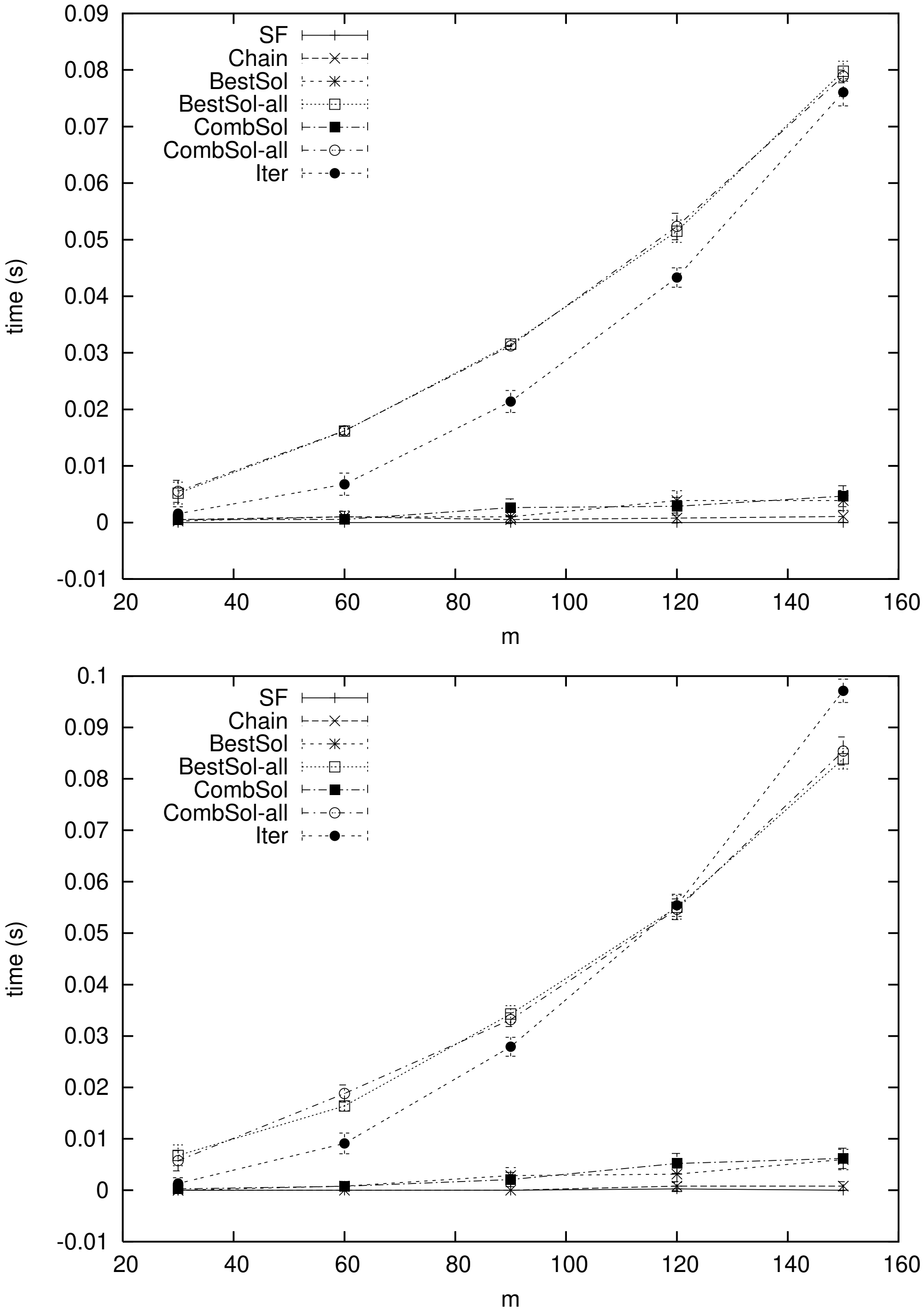

Figure 2.5: Time performance of algorithms for MAXRPC. $n = 16$, m ranges from 30 to 150, $k = 8$. *Top*: uniform distribution. *Bottom*: Gaussian distribution.

2.5 Conclusions

We have presented various algorithms for the MAXPC and MAXRPC problems in rings and have demonstrated results concerning the achieved practical performance.

To evaluate the experimental results we take into consideration the number of satisfied requests as well as the time performance. Taking into account both measures we first remark that MAX(R)PC-CombSol is probably the algorithm of choice for practical purposes, since it achieves one of the best performances with respect to the number of satisfied requests, and at the same time its time requirements are relatively low. Of course MAX(R)PC-Iter and MAX(R)PC-CombSol-all produce better solutions than MAX(R)PC-CombSol, especially as k increases. Undoubtedly, MAX(R)PC-Iter performs better than all other algorithms for very large k but it gets much slower at the same time, because its time complexity depends linearly on k.

From a practical point of view, we can assume that the number of nodes and the number of wavelengths are fixed, thus it is important to consider the behavior of algorithms with respect to the number of requests m. The superiority of MAX(R)PC-CombSol is even more clear in this case considering its time performance. The next best choice seems to be MAX(R)PC-Iter which exhibits an intermediate time performance.

MAX(R)PC-BestSol and MAX(R)PC-BestSol-all are not at all competitive because they fail to provide better solutions than much faster algorithms such as MAX(R)PC-SF, MAX(R)PC-Chain, and MAX(R)PC-CombSol. The new greedy heuristic MAX(R)PC-SF is a decent choice whenever time is crucial, since it achieves relatively large solutions while being one of the fastest algorithms. Tables 2.1 and 2.2 summarize the above observations.

Directions for further research include fine-tuning of some parts of the algorithms. For example, it would make sense to set a threshold on the number of iterations of MAX(R)PC-Iter and combine it with some other strategy for the remaining colors; this could result in more acceptable running times even for large values of k.

Table 2.1: An empirical ranking of the algorithms for problems MaxPC and MaxRPC with respect to their performance in the experiments in terms of number of satisfied requests.

Algorithm	Request satisfaction
Max(R)PC-Iter [75]	★★★★★
Max(R)PC-CombSol-all	★★★★★
Max(R)PC-CombSol [64]	★★★★
Max(R)PC-SF [10]	★★
Max(R)PC-BestSol-all	★★
Max(R)PC-Chain [22]	★
Max(R)PC-BestSol [63]	★

Table 2.2: An empirical ranking of the algorithms for problems MaxPC and MaxRPC with respect to their performance in the experiments in terms of time efficiency.

Algorithm	Time efficiency
Max(R)PC-CombSol [64]	★★★★★
Max(R)PC-SF [10]	★★★★★
Max(R)PC-Chain [22]	★★★★★
Max(R)PC-BestSol [63]	★★★★★
Max(R)PC-Iter [75]	★★★
Max(R)PC-CombSol-all	★
Max(R)PC-BestSol-all	★

Chapter 3

Maximum Profit Wavelength Assignment in WDM Rings

3.1 Introduction

In this chapter we present four algorithms for the MaxProfit-PC problem in rings with undirected requests. Our algorithms combine ideas from algorithms for MaxPC [64, 75] with new techniques specially designed for coloring paths with profits. We give theoretical bounds on the approximation ratio achieved by these algorithms and then move on to perform an experimental comparison with respect to the total profit of the solutions they produce and the execution time they require [11, 12].

One of the results of this comparison is that Match-and-Replace, a novel algorithm that we propose, performs only marginally worse than Iterative, which is based on a well-known technique and gives the best theoretical guarantee for the approximation ratio among the implemented algorithms. At the same time, Match-and-Replace is several orders of magnitude faster than Iterative. A second finding of the experimental comparison is that a natural greedy heuristic with non-constant theoretical approximation guarantee actually performs quite competently and is also exceptionally fast.

While MaxPC, the cardinality version of the problem, has been studied by several researchers [75, 32, 64, 21, 22], MaxProfit-PC has been considered in rather few papers [21, 22]. Both MaxProfit-PC and MaxPC are NP-hard even in simple networks such as rings and trees; this can be shown by an immediate reduction from the corresponding color minimization problem (see e.g. [75]).

MaxProfit-PC in chains is also known as the "weighted k-coloring of intervals" problem, which can be solved exactly in polynomial time as shown by Carlisle and Lloyd [22]. In the case of MAXPROFIT-PC in rings, Caragiannis [21] has presented a randomized algorithm based on linear programming that achieves an expected approximation ratio of 0.67. Let us note here that, although the algorithm in [21] achieves a slightly better worst-case approximation ratio than the algorithms presented in this chapter, we have chosen not to include it in our experimental comparison since our focus is on deterministic and purely combinatorial algorithms.

3.1.1 Preliminaries

Let $w : \mathcal{P} \to \mathbb{Q}^+$ be a function assigning positive rational weights to the paths in some set $\mathcal{P}$. For any $A \subseteq \mathcal{P}$, we will employ the notation $w(A)$ for the total weight of A: $w(A) = \sum_{p \in A} w(p)$. Similarly, for any set $\mathcal{S}$ of subsets of $\mathcal{P}$, we will employ the notation $w(\mathcal{S})$ for the sum of total weights of the elements of $\mathcal{S}$: $w(\mathcal{S}) = \sum_{A \in \mathcal{S}} w(A)$. Note that, if $\mathcal{S}$ contains mutually disjoint subsets of $\mathcal{P}$, then $w(\mathcal{S}) = w\left(\bigcup_{A \in \mathcal{S}} A\right)$.

We denote by $|p|$ the number of edges of path p. Given a set of paths $\mathcal{P}$ and a coloring thereof, the subset of $\mathcal{P}$ that is colored with color a_i is called the *i-th color class* of $\mathcal{P}$ and is denoted by $\mathcal{P}(i)$. We will also use the notation $\mathcal{P}^q$ for the subset of $\mathcal{P}$ that overlaps with path q, and $\mathcal{P}^{\neg q}$ for $\mathcal{P} \setminus \mathcal{P}^q$.

Carlisle and Lloyd [22] give an exact algorithm for MAXPROFIT-PC in chains that runs in $O(km \log m)$ time. In the sequel, we will often use this algorithm as a subroutine for the algorithms that we present. We will refer to this algorithm as the "Carlisle-Lloyd algorithm".

3.2 Match and Replace

We propose a novel algorithm for MAXPROFIT-PC in rings. The Match-and-Replace algorithm is based on a popular technique used for rings, namely to pick a separation edge and remove it from the ring. The set of requests is then partitioned, with respect to the separation edge, into two subsets: the subset of requests that use the separation edge, and the subset of requests that do not use it. Observe that the latter subset can be regarded as an instance of MAXPROFIT-PC in a chain, and thus it can be colored optimally in polynomial time. After this step, the algorithm tries to color some of the requests that use the separation edge, possibly sacrificing some of the requests that have already been colored. To that end, it computes a

Algorithm 5 Match-and-Replace

Input: an instance $\langle G, \mathcal{P}, w, k\rangle$ of MaxProfit-PC, where G is a ring

1: Pick an arbitrary separation edge e of the ring; let $\mathcal{P}_e$ be the set of paths that use edge e and $\mathcal{P}_c = \mathcal{P} \setminus \mathcal{P}_e$.

2: Color the instance $\langle G - e, \mathcal{P}_c, w, k\rangle$ optimally, using the Carlisle-Lloyd algorithm for MaxProfit-PC in chains.

3: Let $\mathcal{P}_c(i)$ be the i-th color class of $\mathcal{P}_c$, $1 \le i \le k$ (note that some color classes may be empty).

4: Construct the weighted path compatibility graph H that corresponds to the separation edge picked in Step 1 and the partial coloring obtained in Step 2.

5: Find a maximum-weight matching M of H.

6: **for all** edges $(\mathcal{P}_c(i), q) \in M$ **do**

7: Uncolor all paths in $\mathcal{P}_c(i)^q$ and color path $q \in \mathcal{P}_e$ with color a_i.

8: **end for**

maximum-weight matching on the corresponding *weighted path compatibility graph.*

Definition 3.1 (Weighted path compatibility graph). *Let $\langle G, \mathcal{P}, w, k\rangle$ be an instance of* MaxProfit-PC *where G is a ring, and let e be a separation edge partitioning the path set into $\mathcal{P}_e$ and $\mathcal{P}_c = \mathcal{P} \setminus \mathcal{P}_e$ where $\mathcal{P}_e$ is the set of paths using edge e. For any partial coloring of the paths in $\mathcal{P}_c$, the corresponding weighted path compatibility graph is a weighted complete bipartite graph $H = (U, E)$, where*

$$U = \{\mathcal{P}_c(i) : 1 \le i \le k\} \cup \mathcal{P}_e \tag{3.1}$$

and edge weights $h : E \to \mathbb{Q}^+$ are defined as follows:

$$h\left(\mathcal{P}_c(i), q\right) = w(q) - w\left(\mathcal{P}_c(i)^q\right) \quad . \tag{3.2}$$

A detailed description of the algorithm is presented in Algorithm 5. We prove below that this algorithm achieves an approximation ratio of $\frac{1}{2}$, and that the analysis that we provide is tight.

Theorem 3.2. *Match-and-Replace is a $\frac{1}{2}$-approximation algorithm for the* MaxProfit-PC *problem in rings.*

Proof. Let OPT be the value of any optimal solution of the ring instance, OPT_c be the value of any optimal solution of the instance constrained to path set $\mathcal{P}_c$, and OPT_e be the value of any optimal solution of the instance constrained to path set $\mathcal{P}_e$. Because $\mathcal{P}_e$ and $\mathcal{P}_c$ form a partition of $\mathcal{P}$,

$$\mathrm{OPT} \leq \mathrm{OPT}_c + \mathrm{OPT}_e \ . \tag{3.3}$$

Let SOL_c be the value of the solution obtained in Step 2 of the algorithm (chain subinstance solution), and SOL be the value of the final solution. Clearly,

$$\mathrm{SOL} = \mathrm{SOL}_c + h(M) \tag{3.4}$$

where $h(M)$ is the sum of the weights of the edges that belong to the matching M computed in Step 5 (recall that h is the edge weight function of the weighted path compatibility graph H). The instance $\langle G - e, \mathcal{P}_c, w, k\rangle$ is solved optimally in Step 2. Therefore, taking also into account Equation 3.4 we have that:

$$\mathrm{OPT}_c = \mathrm{SOL}_c \leq \mathrm{SOL} \ . \tag{3.5}$$

Let $\mathcal{S} = \{\mathcal{P}_c(i) : 1 \leq i \leq k\}$, and $\mathcal{S}_M$ be the set of $\mathcal{P}_c(i)$'s that are matched by M. Similarly, let $\mathcal{P}_{e,M}$ be the paths in $\mathcal{P}_e$ that are matched by M. Finally, let K be the set of the k most profitable paths of $\mathcal{P}_e$. We will now show that

$$\mathrm{OPT}_e = w(K) \leq \mathrm{SOL} \ . \tag{3.6}$$

For the sake of analysis we will examine a solution SOL' that Match-and-Replace would have computed if it had chosen a matching M' of a subgraph H' of H in Step 5. The bipartite graph H' has the same node set and the same edge weight function as H, but only a subset of the edges of H. More specifically, for every pair $(\mathcal{P}_c(i), q)$: the edge $(\mathcal{P}_c(i), q)$ is in H' if and only if $w(q) - w(\mathcal{P}_c(i)) > 0$ and $q \in K$. Let M' be a maximum matching in H', and let $\mathcal{S}_{M'}$ and $\mathcal{P}_{e,M'}$ be defined analogously for M' as for M. Similarly to Equation 3.4,

$$\mathrm{SOL}' = \mathrm{SOL}_c + h(M') \ . \tag{3.7}$$

Note that $\mathrm{SOL}_c = w(\mathcal{S})$. We have:

$$\begin{aligned} h(M') &= w(\mathcal{P}_{e,M'}) - \sum_{(P(i),q)\in M'} w(\mathcal{P}_c(i)^q) && (3.8)\\ &= w(\mathcal{P}_{e,M'}) - \sum_{(\mathcal{P}_c(i),q)\in M'} (w(\mathcal{P}_c(i)) - w(\mathcal{P}_c(i)^{\neg q})) && (3.9)\\ &= w(\mathcal{P}_{e,M'}) - w(\mathcal{S}_{M'}) + \sum_{(\mathcal{P}_c(i),q)\in M'} w(\mathcal{P}_c(i)^{\neg q}) \ . && (3.10) \end{aligned}$$

Equation 3.7 may then be rewritten as follows:

$$\mathrm{SOL}' = w(\mathcal{S} \setminus \mathcal{S}_{M'}) + w(\mathcal{P}_{e,M'}) + \sum_{(\mathcal{P}_c(i),q)\in M'} w(\mathcal{P}_c(i)^{\neg q}) \ . \tag{3.11}$$

We observe that $\mathcal{P}_{e,M'} \subseteq K$ and therefore $w(\mathcal{P}_{e,M'}) + w(K \setminus \mathcal{P}_{e,M'}) = w(K)$, so the last sum can be expanded in the following way:

$$\mathrm{SOL}' = w(\mathcal{S} \setminus \mathcal{S}_{M'}) + w(K) - w(K \setminus \mathcal{P}_{e,M'}) + \sum_{(\mathcal{P}_c(i),q) \in M'} w(\mathcal{P}_c(i)^{\neg q}) \quad . \tag{3.12}$$

Observe also that for any $\mathcal{P}_c(i) \notin \mathcal{S}_{M'}$ and $q \notin \mathcal{P}_{e,M'}$, there must be no edge between them in H', hence $w(\mathcal{P}_c(i)) \geq w(q)$. Moreover, $w(\mathcal{S} \setminus \mathcal{S}_{M'})$ and $w(K \setminus \mathcal{P}_{e,M'})$ are sums with the same number of terms because $|K| = |\mathcal{S}| = k$ and $|\mathcal{S}_{M'}| = \left|\mathcal{P}_{e,M'}\right|$. These observations imply that $w(\mathcal{S} \setminus \mathcal{S}_{M'}) - w(K \setminus \mathcal{P}_{e,M}) \geq 0$, therefore Equation 3.12 yields:

$$\mathrm{SOL}' \geq w(K) \quad . \tag{3.13}$$

Since H' is a subgraph of H, M' is a matching also for H, although probably not a maximum-weight one. Therefore, $h(M) \geq h(M')$, which implies, from Equations 3.4 and 3.7, that $\mathrm{SOL} \geq \mathrm{SOL}'$. Combining this last inequality with Equation 3.13, we obtain Equation 3.6.

By Equations 3.5 and 3.6, SOL is an upper bound on both OPT_e and OPT_c, which together with Equation 3.3 gives:

$$\mathrm{SOL} \geq \frac{\mathrm{OPT}}{2} \quad . \tag{3.14}$$

□

Example 3.3 (Tight example for the approximation ratio of Match-and-Replace). *Consider the* MAXPROFIT-PC *instance illustrated in Figure 3.1. There is only one available color and three paths p_1, p_2, and p_3. Paths p_1 and p_3 are non-overlapping, while p_2 overlaps with both p_1 and p_3. The profits of the paths are: $w(p_1) = w(p_3) = a$ and $w(p_2) = a + 1$, where a is an arbitrary value. Assuming that edge e, as shown in Figure 3.1, is picked as separation edge in Step 1, it is straightforward to verify that Match-and-Replace will color path p_2 with the only available color, while the optimal solution would be to color paths p_1 and p_3. Therefore, the profit of the solution returned by the algorithm can be as bad as a fraction $\frac{a+1}{2a}$ of the optimal, which approaches $\frac{1}{2}$ as a goes to infinity.*

Time complexity of Match-and-Replace The most time-consuming part of the algorithm is the maximum-weight matching computation of Step 5. The compatibility graph H has $O(k + m)$ nodes and $O(km)$ edges; recall that m is the number of paths and k is the number of available colors in the original instance. Therefore, Step 5 takes $O(km(k + m) + (k + m)^2 \log(k + m))$ time. Under the reasonable assumption that $k \ll m$, the time complexity of the algorithm becomes $O(m^2(k + \log m))$.

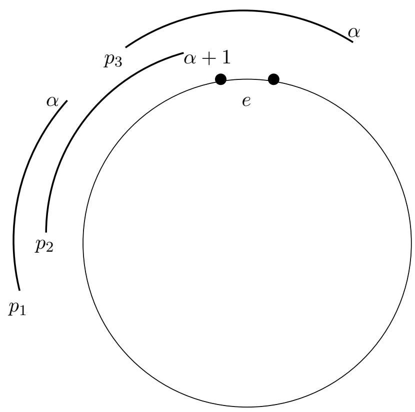

Figure 3.1: An instance of MaxProfit-PC in which the Match-and-Replace algorithm performs as badly as possible. There is only one available color and three paths, p_1, p_2, and p_3 with profits a, $a + 1$, and a respectively. Assuming that Match-and-Replace picks edge e as separation edge in Step 1, it will color path p_2 for a profit of $a + 1$, while the optimal solution would be to color paths p_1 and p_3 for a profit of $2a$. The value of a is arbitrary.

3.3 Other Approaches for Approximating MaxProfit-PC

In this section we present three more algorithms for MaxProfit-PC, which we call Best-Choice, Iterative, and MPLU-Greedy.

3.3.1 Best Choice

A second, more naive application of the separation edge technique involves picking the best of the following two solutions:

1. the solution obtained by coloring optimally the paths that do not use the separation edge, and

2. using one color for each of the k most profitable paths that use the separation edge.

We call this algorithm Best-Choice and a detailed description is given in Algorithm 6. We prove that this algorithm also achieves an approximation ratio of $\frac{1}{2}$ for MaxProfit-PC in rings.

Theorem 3.4. *Best-Choice is an $\frac{1}{2}$-approximation algorithm for* MaxProfit-PC *in rings.*

Algorithm 6 Best-Choice

Input: an instance $\langle G, \mathcal{P}, w, k\rangle$ of MAXPROFIT-PC, where G is a ring

1: Pick an arbitrary separation edge e of the ring; let $\mathcal{P}_e$ be the set of paths that use edge e and $\mathcal{P}_c = \mathcal{P} \setminus \mathcal{P}_e$.
2: Color the instance $\langle G - e, \mathcal{P}_c, w, k\rangle$ optimally, using the Carlisle-Lloyd algorithm for MAXPROFIT-PC in chains. Let k' be the number of colors used in this partial coloring. Use the remaining colors, if any, for the $k - k'$ most profitable paths in $\mathcal{P}_e$. Let $\mathcal{P}_A$ be the set of colored paths.
3: Let $\mathcal{P}_B$ be the set of the k most profitable paths in $\mathcal{P}_e$.
4: **if** $w(\mathcal{P}_A) > w(\mathcal{P}_B)$ **then** return the coloring obtained in Step 2 for $\mathcal{P}_A$.
5: **else** return the coloring that uses a different color for each path in $\mathcal{P}_B$.
6: **end if**

Proof. Let OPT be the value of any optimal solution of the ring instance, OPT_c be the value of any optimal solution of the instance constrained to path set $\mathcal{P}_c$ and OPT_e be the value of any optimal solution of the instance constrained to path set $\mathcal{P}_e$. Since $\mathcal{P}_c$ and $\mathcal{P}_e$ form a partition of $\mathcal{P}$,

$$\mathrm{OPT} \leq \mathrm{OPT}_c + \mathrm{OPT}_e \ . \tag{3.15}$$

By Step 2 of the algorithm, $w(\mathcal{P}_A) \geq OPT_c$. Moreover, by Step 3 of the algorithm, $w(\mathcal{P}_B) = OPT_e$. Given the fact that $\mathrm{SOL} = \max\{w(\mathcal{P}_A), w(\mathcal{P}_B)\}$, it follows that:

$$\mathrm{OPT}_c \leq w(\mathcal{P}_A) \leq \mathrm{SOL} \ , \tag{3.16}$$

and

$$\mathrm{OPT}_e = w(\mathcal{P}_B) \leq \mathrm{SOL} \ . \tag{3.17}$$

Combining Equations 3.15, 3.16, and 3.17, we get that

$$\mathrm{SOL} \geq \frac{\mathrm{OPT}}{2} \ . \tag{3.18}$$

□

Furthermore, observe that the MAXPROFIT-PC instance illustrated in Figure 3.1 also serves as a tight example for the Best-Choice algorithm.

Time complexity of Best-Choice Step 2 requires $O(km \log m)$ time. The selection of the k most profitable paths in Step 3 can be done in $O(m)$ time, by selecting the path with the $(|\mathcal{P}_e| - k)$-th smallest profit using a known linear time selection algorithm which at the same time performs a partition with the selected element as pivot (see e.g. [25, p. 189]). Therefore, the overall time complexity of the algorithm is dominated by Step 2 and is $O(km \log m)$.

Algorithm 7 Iterative

Input: an instance $\langle G, \mathcal{P}, w, k\rangle$ of MAXPROFIT-PC, where G is a ring

for all colors a_i **do**
  $\mathcal{S}_i := \emptyset$
  for all paths $p \in \mathcal{P}$ **do**
    $\mathcal{S}_p := \{p\}$
    Find a maximum-profit set of edge-disjoint paths that do not overlap with p by running the Carlisle-Lloyd algorithm for MAXPROFIT-PC on the instance $\langle G - p, \mathcal{P}^{\neg p}, w, 1\rangle$, where $G - p$ is the graph obtained by removing all edges of path p from G; insert these paths in $\mathcal{S}_p$.
    if $w(\mathcal{S}_p) > w(\mathcal{S}_i)$ **then**
      $\mathcal{S}_i := \mathcal{S}_p$
    end if
  end for
  Color all requests in $\mathcal{S}_i$ with color a_i.
  $\mathcal{P} := \mathcal{P} \setminus \mathcal{S}_i$
end for

3.3.2 Iterative

In this section we present an algorithm that iteratively colors a maximum-profit subset of non-overlapping requests. This algorithm is based on a known maximum coverage technique that also applies to coloring problems (see e.g. Wan and Liu [75], Erlebach et al. [35], or Awerbuch et al. [6]). We will refer to this algorithm as Iterative. Iterative works as follows: during each iteration i it computes for each path p a maximum-profit subset $\mathcal{S}_p$ of non-overlapping paths that contains p. Finally, the set $\mathcal{S}_p$ with maximum profit is colored with color a_i and is removed from $\mathcal{P}$.

In order to compute $\mathcal{S}_p$ it suffices to solve MAXPROFIT-PC in the following instance: $\langle G - p, \mathcal{P}^{\neg p}, w, 1\rangle$, where $G - p$ is the graph obtained by removing all edges of path p from G. Observe that this instance is a chain instance and can be solved optimally with the Carlisle-Lloyd algorithm. The solution of this instance, together with path p, constitutes the set $\mathcal{S}_p$. We give a detailed description of the algorithm in Algorithm 7.

It has been observed by Erlebach et al. [35] that a straightforward adaptation of the technique of Awerbuch et al. [6] can be used to prove that the Iterative algorithm achieves an approximation ratio of $1 - \frac{1}{e}$ for the cardinality version of MAXPROFIT-PC, where all requests have profit equal to 1. It turns out that the analysis goes through for the case of non-uniform profits as well. We present the proof below for the sake of completeness.

Theorem 3.5. *Iterative is an* $\left(1 - \frac{1}{e}\right)$*-approximation algorithm for* MaxProfit-PC *in rings.*

Proof. Let $\langle G, \mathcal{P}, w, k\rangle$ be an input to the Iterative algorithm, and $t_i = w(\mathcal{S}_i)$, $1 \le i \le k$, be the total profit of the paths colored with color a_i during the i-th iteration of the algorithm. Let OPT be the total profit of an optimal solution.

We first prove that, for any $j : 1 \le j \le k$:

$$\sum_{i=1}^{j} t_i \ge \mathrm{OPT} \cdot \left(1 - \left(1 - \frac{1}{k}\right)^j\right) \ . \tag{3.19}$$

Equation 3.19 certainly holds for $j = 1$: there is at least one set of non-overlapping edge-disjoint paths with total profit at least $\frac{\mathrm{OPT}}{k}$, and the Iterative algorithm finds the largest such set during the first iteration. Assuming that Equation 3.19 holds for $j = s - 1$, we get:

$$\begin{aligned}
\sum_{i=1}^{s} t_i &= \sum_{i=1}^{s-1} t_i + t_s && (3.20)\\
&\ge \sum_{i=1}^{s-1} t_i + \frac{\mathrm{OPT} - \sum_{i=1}^{s-1} t_i}{k} && (3.21)\\
&= \left(1 - \frac{1}{k}\right) \cdot \sum_{i=1}^{s-1} t_i + \frac{\mathrm{OPT}}{k} && (3.22)\\
&\ge \left(1 - \frac{1}{k}\right) \cdot \mathrm{OPT} \cdot \left(1 - \left(1 - \frac{1}{k}\right)^{s-1}\right) + \frac{\mathrm{OPT}}{k} && (3.23)\\
&= \mathrm{OPT} \cdot \left(1 - \left(1 - \frac{1}{k}\right)^{s}\right) \ . && (3.24)
\end{aligned}$$

Therefore, Equation 3.19 holds for all j between 1 and k. By setting $j = k$ in Equation 3.19 we get:

$$\sum_{i=1}^{k} t_i \ge \mathrm{OPT} \cdot \left(1 - \left(1 - \frac{1}{k}\right)^k\right) \ge \left(1 - \frac{1}{e}\right) \cdot \mathrm{OPT} \ , \tag{3.25}$$

that is, the solution returned by the Iterative algorithm is at least a fraction of $1 - \frac{1}{e} \approx 0.632$ of the optimal. □

Time complexity of Iterative The time complexity of Step 5 of the algorithm is $O(km \log m)$, and in the worst case at most km iterations of the inner loop are needed. Therefore, the total time complexity of Iterative is $O(k^2 m^2 \log m)$.

Algorithm 8 MPLU-Greedy

Input: an instance $\langle G, \mathcal{P}, w, k\rangle$ of MAXPROFIT-PC

1: Sort the paths $p \in \mathcal{P}$ in order of non-increasing ratio $\frac{w(p)}{|p|}$.
2: **for all** paths $p \in \mathcal{P}$ (in the order of Step 1) **do**
3: If there is some color a_i that can be assigned to p, color path p with color a_i.
4: **end for**

3.3.3 Greedy

We present a natural greedy heuristic for MAXPROFIT-PC. The key idea is that the more edges a path uses, the more likely it is to block other, possibly more profitable paths from being added to the solution. On the other hand, a path may be so profitable that it is worth picking it in the solution, despite its length. Translating these observations into an algorithm, we end up with the following approach: consider the paths in non-increasing order of the ratio of their profit over their length; if there is an available color for the current path, color it—otherwise drop this path. We call this algorithm Most Profit per Length Unit Greedy, for short MPLU-Greedy (Algorithm 8). It is very fast and easy to implement but, as we show below, there is no constant ρ, $0 < \rho \leq 1$, such that the profit of the solution returned by the algorithm is guaranteed to be at least a fraction ρ of the optimal. Note that the algorithm works in any network topology, not just in rings.

Example 3.6 (Non-constant approximation ratio of MPLU-Greedy in rings)**.** *Consider the instance of* MAXPROFIT-PC *that is illustrated in Figure 3.2. There is only one available color, and two overlapping paths, p_1 and p_2, with $w(p_1) = \ell - 1$ and $w(p_2) = 1$. The length of the paths p_1 and p_2 is ℓ and 1, respectively. The MPLU-Greedy algorithm will first consider path p_2 and color it with the only available color. This will result in the path p_1 remaining uncolored. The total profit of this solution is 1. On the other hand, the optimal solution would use the only available color to color path p_1 and obtain a profit of $\ell - 1$. This implies that the solution returned by the algorithm can be as bad as a fraction $\frac{1}{\ell-1}$ of the optimal. Given that ℓ can be arbitrarily large, the algorithm can be made to perform arbitrarily badly.*

Time complexity of MPLU-Greedy A simple implementation of the algorithm requires $\mathcal{O}(nmk)$ time.

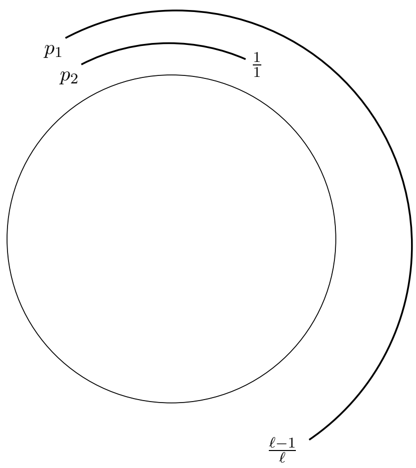

Figure 3.2: An instance of MAXPROFIT-PC in which the MPLU-Greedy algorithm performs badly. There is only one available color and two paths, p_1 and p_2 with profits $\ell - 1$ and 1 respectively, and length ℓ and 1 respectively. The MPLU-Greedy algorithm will color path p_2 for a profit of 1, while the optimal solution would be to color path p_1 for a profit of $\ell - 1$. The value of ℓ is arbitrary.

3.4 Numerical Results

3.4.1 Experimental Setup

We implemented all algorithms in C++, making use of the LEDA™ class library of efficient data types and algorithms. All source files were compiled with the Borland™ C++ 5.5 for Win32 compiler, set to generate fastest possible code. We relied on LEDA routines and classes for graph, array, list and priority queue operations including sorting and finding maximum-weight matchings in bipartite graphs. The experiments were run on a Pentium™ 4 clocked at 3.2GHz with 512MB of memory.

Instance packs An *instance pack* is a set of 50 randomly generated instances $\langle G, \mathcal{P}, w, k\rangle$ of MAXPROFIT-PC, where G is a ring, specified by the following parameters:

- the number n of nodes in the ring,
- the number m of requests in the set $\mathcal{P}$,
- the number k of available colors,
- an upper bound W on the profit of the requests, and

- the manner in which paths are generated, specified either as *uniform* or as *gaussian:μ:σ*.

Each instance in the instance pack is defined on a ring with n nodes. There are k available colors. The path set $\mathcal{P}$ of the instance has cardinality m, and the profit of each path in $\mathcal{P}$ is selected uniformly at random from the set $\{1, \ldots, W\}$. The path itself is generated in one of two ways:

- If the mode of generation is *uniform*, the two endpoints of the path are selected independently uniformly at random from the node set of the ring. The edges actually used by the path are the edges that connect the first endpoint to the second one, in the clockwise direction.

- If the mode of generation is *gaussian:μ:σ*, then the first endpoint of the path is selected uniformly at random from the node set of the ring. Subsequently, the length of the path is selected at random, following the normal distribution with mean μ and standard deviation σ. The path spans as many edges as its length in the clockwise direction, starting from the first endpoint.

For each instance pack that we generated, we executed each algorithm on all instances of the pack and measured the average execution time and the average profit of satisfied requests. Furthermore, for each of these values we calculated a 95 percent confidence interval which is shown on the plots.

In each one of the figures discussed below (Figures 3.3, 3.4, 3.5, 3.6, and 3.7), we present the results corresponding to several instance packs. In each of these instance packs, we keep four of the above parameters fixed and let one of them vary in order to exhibit the effect of this parameter on the execution time and on the profit of satisfied requests.

Note that execution times were measured using the *timer* class of the LEDA package, which does not provide routines for measuring exact processor time. However, we ran the experiments on a dedicated machine and kept background processes at a minimum.

Computing an upper bound on OPT In order to obtain an estimation of the performance of our algorithms we use the following upper bound on the value of an optimal solution:

$$\mathrm{OPT} \leq \min_{e \in E}\{\mathrm{OPT}_e + \mathrm{OPT}_c\} \ , \tag{3.26}$$

where OPT_e is the total profit of the k most profitable paths using edge e, and OPT_c is the optimal solution of the MAXPROFIT-PC instance that contains only the paths that *do not* use edge e. The latter is computed using the Carlisle-Lloyd algorithm, as discussed earlier.

3.4.2 Discussion

A first observation is that all algorithms perform considerably better than their theoretical guarantee. Indeed, we have included a curve showing the computed upper bound (UB) in our figures and it turns out that all algorithms manage to satisfy a good fraction of an optimal solution, often much better than the theoretically predicted.

In the experiments of Figure 3.3, we compare all algorithms for variable number of nodes ranging from 4 to 16 (typical values for SONET rings), with the number of requests being ten times the number of nodes. The number of available wavelengths is fixed to 8. The endpoints of each request are chosen uniformly at random. The profit of each request is chosen uniformly at random from $\{1, \ldots, 10\}$. We observe that Iterative achieves the best performance, closely followed by Match-and-Replace which is a remarkably faster algorithm. MPLU-Greedy performs quite well, although there is no constant bound on its approximation ratio. Best-Choice has no particular merits, but serves as a good basis in order to exhibit the improvement achieved by Match-and-Replace. In Figure 3.4, experiments with a wider variance of profits than the ones in Figure 3.3, namely between 1 and 100, result in a similar ranking of the algorithms in terms of achieved profit. Observe that, in some cases in Figure 3.4, Match-and-Replace even outperforms the Iterative algorithm (for example in the instance pack with $m = 350$).

In the experiments of Figure 3.5 the load is comparable to the number of wavelengths and this explains the good performance of all algorithms (except Best-Choice). In particular, the average load is around 20 for every 100 requests and thus there are enough wavelengths to color almost all paths, especially for number of requests up to 300.

In the experiments of Figure 3.6 only one endpoint of each request is chosen uniformly at random and the other endpoint is determined in such a way so that the length of the request follows the normal distribution with mean 8 and standard deviation 1. In these experiments MPLU-Greedy appears competent and Iterative displays noticeable superiority. This behavior can be explained if we take into account that the length of paths is about half the cycle and thus with high probability each color can be used for at most two paths. This fact favors Iterative, which has fewer

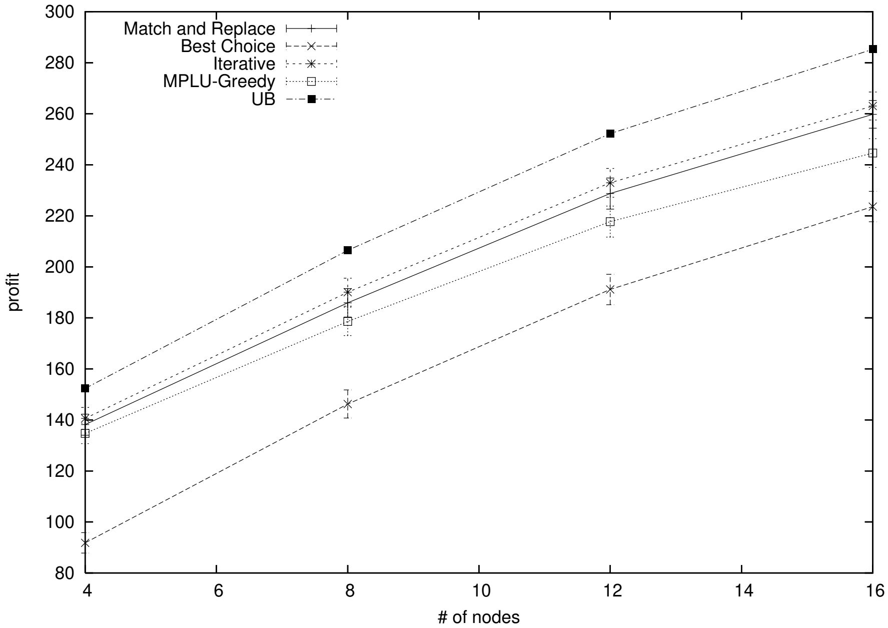

Figure 3.3: Instance pack parameters: n ranges from 4 to 16, $m = 10n$, $k = 8$, $W = 10$, endpoints: *uniform*.

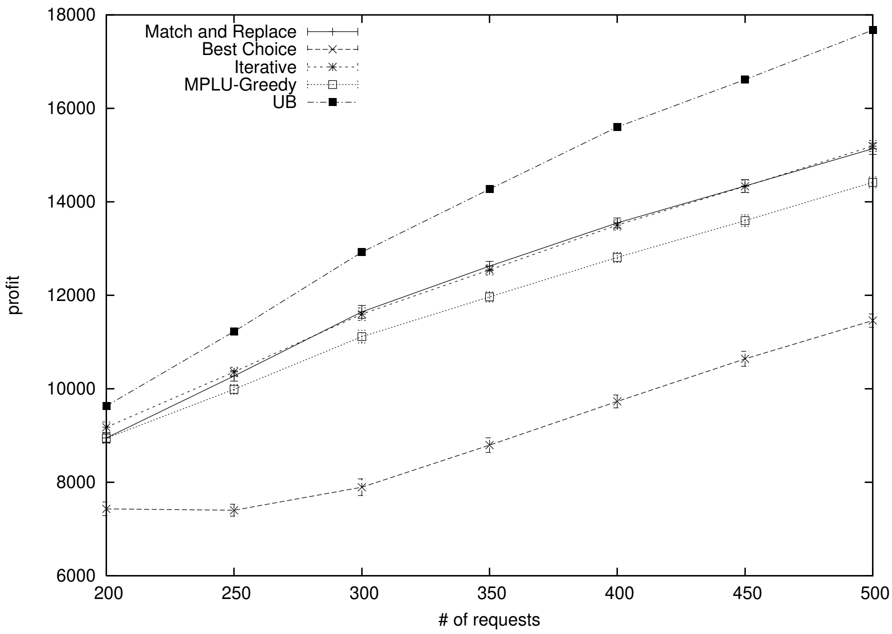

Figure 3.4: Instance pack parameters: $n = 100$, m ranges from 200 to 500, $k = 80$, $W = 100$, endpoints: *uniform*.

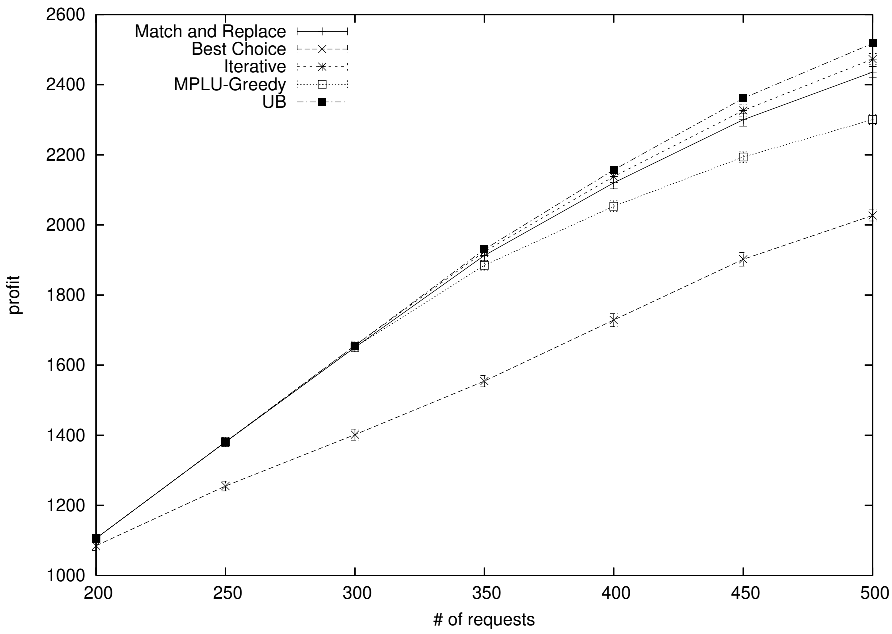

Figure 3.5: Instance pack parameters: n = 100, m ranges from 200 to 500, k = 80, W = 10, endpoints: *gaussian*:20:2.

limitations on the path combinations it tries.

Figure 3.7 illustrates a comparison of the running time of the algorithms. We observe that Iterative is thousands of times slower than Match-and-Replace, which has comparable performance in terms of achieved profit. Best-Choice is somewhat faster than Match-and-Replace. MPLU-Greedy is several times faster than Best-Choice.

3.5 Conclusions

To evaluate the experimental results we take into consideration the obtained profit as well as the time performance. Taking into account both measures we first remark that Match-and-Replace should be the algorithm of choice for practical purposes, since it achieves one of the best performances with respect to the obtained profit, and at the same time its time requirements are reasonably low. In most cases Iterative produces marginally better solutions than Match-and-Replace, but its time consumption could be prohibitive. On the other hand, if time efficiency is crucial it would also make sense to consider MPLU-Greedy, which is a very fast algorithm with acceptable performance. Taking into account both per-

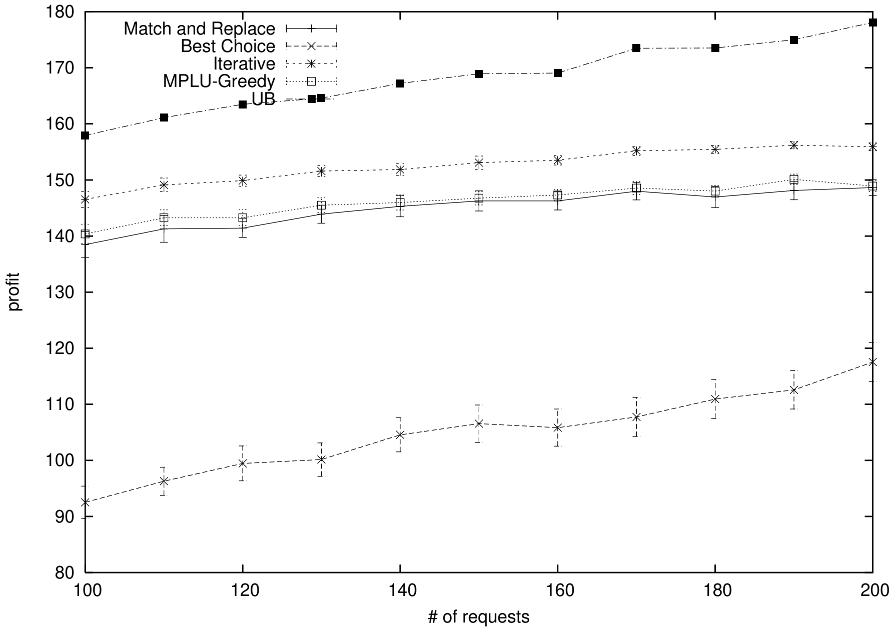

Figure 3.6: Instance pack parameters: $n = 16$, m ranges from 100 to 200, $k = 8$, $W = 10$, endpoints: *gaussian*:8:1.

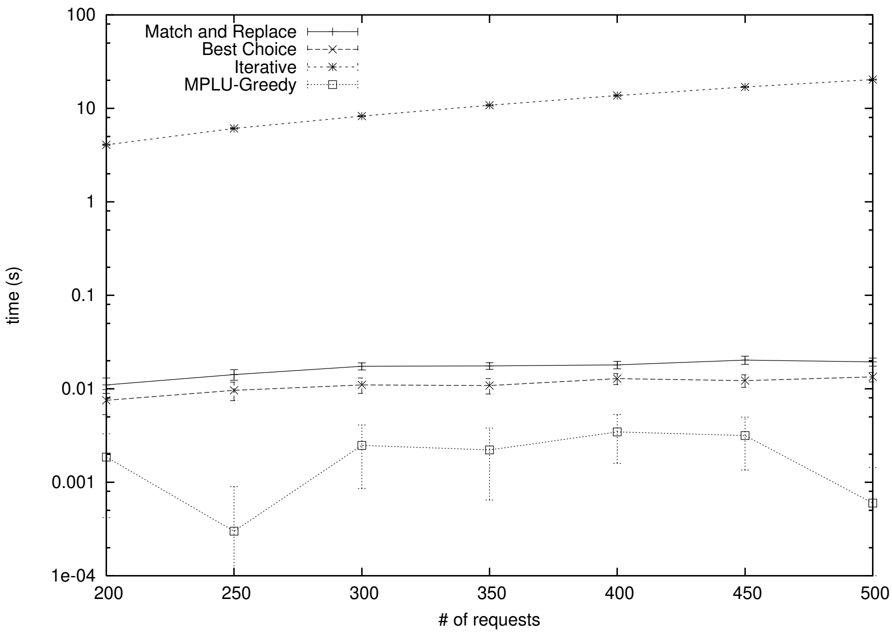

Figure 3.7: Instance pack parameters: $n = 100$, m ranges from 200 to 500, $k = 80$, $W = 100$, endpoints: *uniform*.

Table 3.1: An empirical ranking of the algorithms for problem MaxProfit-PC with respect to their performance in the experiments in terms of attained profit.

Algorithm	Attained profit	Approximation ratio
Iterative	★★★★	0.632
Match-and-Replace	★★★★	0.5
MPLU-Greedy	★★★	non-constant
Best-Choice	★	0.5

Table 3.2: An empirical ranking of the algorithms for problem MaxProfit-PC with respect to their performance in the experiments in terms of time efficiency.

Algorithm	Time efficiency	Time complexity
MPLU-Greedy	★★★★	$O(nmk)$
Match-and-Replace	★★★	$O(m^2(k + \log m))$
Best-Choice	★★★	$O(km \log m)$
Iterative	★	$O(k^2 m^2 \log m)$

formance with respect to profit and time efficiency, as they were assessed from the experimental results, we rank all four algorithms in Tables 3.1 and 3.2. We also include the theoretical bounds on the approximation ratio and time complexity for reference.

Chapter 4

Non-cooperative Wavelength Assignment in Multifiber Optical Networks

4.1 Introduction

The need for efficient access to optical bandwidth has given rise to the study of several optimization problems in the past years. One of the most well-studied among them is the problem of assigning a path and a color (wavelength) to each communication request in such a way that paths of the same color are edge-disjoint and the number of colors used is minimized. Nonetheless, it has become clear that the number of wavelengths in commercially available fibers is rather limited—and will probably remain such in the foreseeable future. Fortunately, the use of multiple fibers has come to the rescue.

In a *multifiber* optical network, a physical link may be implemented with more than one optical fibers deployed in parallel between the endpoins of the link. Naturally, this boosts the available bandwidth. More importantly, it allows for several requests using the same wavelength to be routed on the same physical link, provided that each one uses a different fiber. However, fibers are not unlimited either, therefore it makes sense to minimize their usage. This is particularly interesting from the customer's point of view, for example in situations where one can hire a number of parallel fibers for a certain period and the cost depends on that number.

To this end, several optimization problems have been defined and studied, the objective being to minimize either the maximum fiber multiplicity per edge [4, 3, 2] or the sum of these maximum multiplicities over all edges of the graph [62, 35, 76]; in another scenario the allowed fiber multiplicity per edge is given and the goal is to minimize the number of wavelengths needed [53, 49, 35].

In this chapter we consider a non-cooperative model, where each request is issued by a user who tries to optimize her own fiber usage by selecting the most appropriate wavelength, taking into account the choices of other users. This model is mainly motivated by the fact that centralized control in large scale networks may be either infeasible or impractical. We assume that each user is charged according to the maximum fiber multiplicity that the user's choice incurs. More specifically, a user will be charged according to the maximum number of paths that share an edge with her and use the same wavelength. We consider as *social cost* the maximum fiber multiplicity that appears on any edge of the network. Minimizing this quantity is particularly important in cases where fibers are hired or sold as a whole, hence the maximum number of fibers needed on an edge determines the total cost; further motivation can be found in papers that address the corresponding optimization problem (see e.g. [4, 3, 2]). Here we focus on situations where routing is unique (acyclic topologies) or pre-determined—as happens in many practical settings, for example in cases where there are specific routing constraints such as a requirement to use lightpaths that have been set in advance, or shortest-path routing.

We formulate the above model by defining the class of SELFISH PATH MULTICOLORING (S-PMC) games: a game is defined in terms of a graph, a set of paths, and the number of colors k. Each player controls a path in the graph and has to choose a color for that path from the set of available colors $W = \{a_1, \ldots, a_k\}$. A player is charged according to the maximum multiplicity of her color along her path. We consider as *social cost* the maximum color multiplicity per edge, i.e., the maximum number of paths of same color that use an edge.

It is worth mentioning that path multicoloring problems can be used to model situations that arise in various practical settings not necessarily limited to optical networking. For example, our model may also find applications in communication networks where packets are transmitted using time-division multiplexing. In this case, colors represent timeslots and a color's multiplicity determines the number of frames it takes for each user of the corresponding timeslot to transmit a single packet. Therefore, the social cost of our model is proportional to the total number of frames needed for all users to complete their transmissions, assuming they all possess the same number of packets.

In the rest of this chapter, we first give an overview of related work and then move on to present our results on SELFISH PATH MULTICOLORING games [8]. We show an upper bound on the convergence rate of Nash dynamics for S-PMC games, and observe that the price of stability is always equal to 1. We also show how to efficiently compute a Nash equilibrium of minimum social cost for S-PMC games in rooted trees, i.e. games in which each communication request lies entirely on a simple path from some fixed root node to a leaf. For S-PMC games in stars, we prove that a known approximation algorithm for a related optimization problem actually gives an $\frac{1}{2}$-approximate Nash equilibrium.

For general graphs, we obtain two upper bounds on the price of anarchy: the first, which is not hard to show, is equal to the number of available colors. The second, which requires more involved arguments, is equal to the length of a shortest path with maximum disutility in any worst-case Nash equilibrium. For both bounds we provide matching lower bounds. In fact, we prove that these bounds hold even in trees.

Then, we move on to specific network topologies and show that for S-PMC games in stars the price of anarchy is equal to 2. We also provide constant bounds on the price of anarchy for a broad class of S-PMC games in chains and rings, namely for all games with $L = \Omega(k^2)$, where k is the number of available colors and L is the maximum load among all edges of the network. On the other hand, for any $\varepsilon > 0$ we exhibit a class of S-PMC games in chains (and rings) with $L = \Theta(k^{2-\varepsilon})$ for which the price of anarchy is unbounded.

In order to show our upper bounds, we demonstrate path patterns that must be present in any Nash equilibrium, while for the lower bounds we employ recursive construction techniques.

4.2 Related Work

Arguably, the most important notion in the theory of non-cooperative games is the *Nash equilibrium (NE)* [60], a stable state of the game in which no player has incentive to change strategy unilaterally. A fundamental question in this theory concerns the existence of *pure* Nash equilibria (PNE). For various games [37, 67, 56, 74, 58, 36] it has been shown that a pure Nash equilibrium exists or can be found with the use of potential functions. A standard measure of the worst-case quality of Nash equilibria relative to optimal solutions is the *price of anarchy* (PoA) [46], which has been extensively studied for load balancing games [46, 54] and other problems such as routing and facility location [37, 68]. A second known measure related to Nash equilibria is the *price of stability* (PoS), defined in [5].

SELFISH PATH MULTICOLORING games are closely related to a variation of congestion games [19, 13] where a player's cost is determined by her *maximum* latency instead of the usual cost which is the *sum* of her latencies. Next, we briefly explain the relation of those models to ours.

In [19] the authors study atomic routing games on networks, where each player chooses a path to route her traffic from an origin to a destination node, with the objective of minimizing the maximum congestion on any edge of her path. They show that these games always possess at least one optimal pure Nash equilibrium (hence the price of stability is 1) and that the price of anarchy of the game is determined by topological properties of the network; in particular they show that the price of anarchy is upper-bounded by the length of the longest path in the player strategy sets and lower-bounded by the length of the longest cycle. Some of our results extend to their model, since our model mimics traffic routing in the following sense: we may consider a multigraph, where we replace each edge with k parallel edges, one for each color. Each player's strategy set then consists of k different source-destination paths, corresponding to the k available colors in the original model. A further generalization is the model of Banner and Orda [13], where they introduce the notion of bottleneck games. In this model they allow arbitrary latency functions on the edges and consider both the case of splittable and unsplittable flows. They show existence, convergence and non-uniqueness of equilibria and they prove that the price of anarchy for these games is unbounded. Both models are more general than ours; however our model fits better into the framework of all-optical networks for which we manage to provide, among others, smaller upper bounds on the price of anarchy compared to the ones obtained by [19, 13], as well as a better convergence rate to Nash equilibria.

Selfish path coloring in single fiber all-optical networks has been studied in [16, 15, 44, 57]. Bilò and Moscardelli [16] consider the convergence to Nash equilibria of selfish routing and path coloring games. Later, Bilò et al. [15] considered different information levels of local knowledge that players may have for computing their payments in the same games and give bounds for the price of anrchy in chains, rings and trees. The existence of Nash equilibria and the complexity of recognizing and computing a Nash equilibrium for selfish routing and path colorings games under several payment functions are considered by Georgakopoulos et al. [44]. In [57] upper and lower bounds of the price of anarchy for selfish path coloring with and without routing are presented under functions that charge a player only according to her own strategy.

4.3 Preliminaries

Given an undirected graph $G = (V, E)$, a set $\mathcal{P}$ of simple paths defined on G, and a set $W = \{a_1, \ldots, a_k\}$ of available colors, recall that we use the notation $L(e)$ for the load of edge e, i.e., the number of paths that use edge e, and the notation L for the maximum of these loads, i.e., $L = \max_{e\in E} L(e)$.

Given, additionally, an assignment of a color to each path we define the following notation:

Definition 4.1. *1. $\mu(e, c)$ will denote the multiplicity of color c on edge e, i.e. the number of paths that use edge e and are colored with color c.*

2. μ_e will denote the maximum multiplicity of any color on edge e:

$$\mu_e = \max_{c\in W} \mu(e, c) \ . \tag{4.1}$$

3. $\mu_{\max}$ will denote the maximum multiplicity of any color over all edges:

$$\mu_{\max} = \max_{e\in E} \mu_e \ . \tag{4.2}$$

4. $\mu(p, c)$ will denote the maximum multiplicity of color c over the edges of path p:

$$\mu(p, c) = \max_{e\in p} \mu(e, c) \ . \tag{4.3}$$

It will be clear from the context which specific coloring we are referring to when we use the above notation.

The minimum $\mu_{\max}$ that can be attained by some coloring of the paths in $\mathcal{P}$ will be denoted by μ_{OPT}, i.e:

$$\mu_{\text{OPT}} = \min_{\vec{c}} \mu_{\max} \ , \tag{4.4}$$

where $\vec{c}$ ranges over all possible colorings. We note immediately the following:

Fact 4.2. *No coloring can achieve a $\mu_{\max}$ that is smaller than $\left\lceil \frac{L}{k} \right\rceil$. Thus,*

$$\mu_{\text{OPT}} \geq \left\lceil \frac{L}{k} \right\rceil \ . \tag{4.5}$$

4.3.1 Game-Theoretic Model

We now proceed to define the class of selfish path multicoloring games and subclasses thereof.

Definition 4.3 (Selfish path multicoloring games). *A selfish path multicoloring game is the following strategic game defined in terms of an undirected graph G, a set $\mathcal{P}$ of simple paths defined on G, and an integer $k > 0$:*

- *Players: there is one player for each path in $\mathcal{P}$. For simplicity, we will identify a player i with the corresponding path p_i.*
- *Strategies: a strategy for player i is a color c_i chosen from the set $W = \{a_1, \ldots, a_k\}$ of available colors. We say that color c_i is assigned to path p_i or that path p_i is colored with color c_i. All players share the common set of available strategies W. The strategies chosen by all players will be collectively described by a vector $\vec{c} = (c_1, \ldots, c_{|\mathcal{P}|})$. Vector $\vec{c}$ will be called a strategy profile.*
- *Disutility: given a strategy profile $\vec{c} = (c_1, \ldots, c_{|\mathcal{P}|})$, the disutility $f_i : W^{|\mathcal{P}|} \to \mathbb{N}$ of each player i is defined as follows:*

$$f_i(\vec{c}) = \mu(p_i, c_i) \ . \tag{4.6}$$

We denote this game by $\langle\langle G, \mathcal{P}, k \rangle\rangle$.

Definition 4.4. S-PMC *will denote the class of all selfish path multicoloring games $\langle\langle G, \mathcal{P}, k \rangle\rangle$.*

We will use the notation S-PMC($\mathcal{G}$) to denote a subclass of S-PMC that contains only games satisfying a property $\mathcal{G}$. For example, $\mathcal{G}$ may constrain the graph on which the game is defined to belong to a specific graph class, etc.

Following the standard definition [60], we say that a strategy profile $\vec{c} = (c_1, \ldots, c_{|\mathcal{P}|})$ is a *pure Nash equilibrium* (PNE), or simply *Nash equilibrium* (NE) for our purposes, if for each player i it holds that:

$$f_i(c_1, \ldots, c_i', \ldots, c_{|\mathcal{P}|}) \geq f_i(c_1, \ldots, c_i, \ldots, c_{|\mathcal{P}|}) \ , \tag{4.7}$$

for any strategy $c_i' \in W$. Moreover, following the definition of [23], we say that a strategy profile $\vec{c} = (c_1, \ldots, c_{|\mathcal{P}|})$ is an *ε-approximate Nash equilibrium* if for each player i it holds that:

$$f_i(c_1, \ldots, c_i', \ldots, c_{|\mathcal{P}|}) \geq (1 - \varepsilon) \cdot f_i(c_1, \ldots, c_i, \ldots, c_{|\mathcal{P}|}) \ , \tag{4.8}$$

for any strategy $c_i' \in W$. In a Nash equilibrium, no player has incentive to change strategy unilaterally, while in an ε-approximate Nash equilibrium a unilateral change of strategy may result in reducing the player's cost by no more than a factor of $1 - \varepsilon$.

Definition 4.5 (Blocking edges). *If $\vec{c}$ is a strategy profile for an* S-PMC *game $\langle\!\langle G, \mathcal{P}, w \rangle\!\rangle$ and $p_i \in \mathcal{P}$, we say that an edge $e \in p_i$ is an a_j-blocking edge for p_i, or that it blocks a_j for p_i, if*

$$\mu(e, a_j) \geq f_i(\vec{c}) - 1 \ . \tag{4.9}$$

Furthermore, the $\mu(e, a_j)$ paths that are colored with a_j and use edge e are called a_j-blocking paths for p_i.

Intuitively, an a_j-blocking edge for p_i "blocks" p_i from switching to color a_j because if it did, the new disutility of path p_i would be at least $\mu(e, a_j) + 1 \geq f_i(\vec{c})$, no better than its current choice. The following characterization of the Nash equilibria of S-PMC games is immediate from the definitions:

Property 4.6 (Structural characterization of S-PMC Nash equilibria). *A strategy profile for an* S-PMC *game $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ is a Nash equilibrium if and only if every path $p \in \mathcal{P}$ contains at least one a_j-blocking edge for p, for every color a_j.*

Definition 4.7 (Social cost). *The social cost of a strategy profile $\vec{c}$ for an* S-PMC *game is defined as follows:*

$$\mathrm{sc}(\vec{c}) = \max_{e \in E} \mu_e = \mu_{\max} \ . \tag{4.10}$$

It is straightforward to verify that the social cost of a strategy profile coincides with the maximum player disutility in that profile:

$$\mathrm{sc}(\vec{c}) = \max_{e \in E} \mu_e = \max_{p_i \in \mathcal{P}} f_i(\vec{c}) \ . \tag{4.11}$$

We define $\hat{\mu}$ to be the maximum social cost over all strategy profiles that are Nash equilibria:

$$\hat{\mu} = \max_{\vec{c} \text{ is NE}} \mathrm{sc}(\vec{c}) \ . \tag{4.12}$$

Following the standard definitions [46, 5], the *price of anarchy* (PoA) of a game $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ is the worst-case social cost in a Nash equilibrium divided by μ_{OPT}, i.e.:

$$\mathrm{PoA}(\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle) = \frac{\max_{\vec{c} \text{ is NE}} \mathrm{sc}(\vec{c})}{\mu_{\mathrm{OPT}}} = \frac{\hat{\mu}}{\mu_{\mathrm{OPT}}} \ . \tag{4.13}$$

The *price of stability* (PoS) of a game is the best-case social cost in a Nash equilibrium divided by μ_{OPT}:

$$\mathrm{PoS}(\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle) = \frac{\min_{\vec{c} \text{ is NE}} \mathrm{sc}(\vec{c})}{\mu_{\mathrm{OPT}}} \quad . \tag{4.14}$$

The price of anarchy (resp. stability) of a class of games S-PMC($\mathcal{G}$) is the maximum price of anarchy (resp. stability) among all games in S-PMC($\mathcal{G}$).

4.4 Price of Stability, Existence, and Convergence to Nash Equilibria

In this section we prove that any S-PMC game $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ has at least one Nash equilibrium of optimal social cost. Moreover, we prove that starting from an arbitrary strategy profile, any Nash dynamics converges to a Nash equilibrium in at most $4^{|\mathcal{P}|}$ steps. For our purposes, the *Nash dynamics* is a sequence $\vec{c}_0, \vec{c}_1, \ldots$ of strategy profiles where in each profile $\vec{c}_{i+1}$ exactly one player has a different strategy compared to $\vec{c}_i$, and that player has strictly improved her disutility compared to her disutility in $\vec{c}_i$. In other words, the Nash dynamics is a sequence of cost-improving moves of the players in which no particular order of play or fairness criteria is assumed *a priori*.

For any strategy profile $\vec{c}$, we define a disutility vector $\boldsymbol{D}(\vec{c})$ as follows:

$$\boldsymbol{D}(\vec{c}) = (d_L(\vec{c}), \ldots, d_1(\vec{c})) \quad , \tag{4.15}$$

where $d_i(\vec{c})$ stands for the number of players whose disutility is exactly i (note that the disutility of any player cannot be 0 and cannot be greater than L). We use lexicographic-order arguments similar to those in [19, 13] to show that starting from an arbitrary strategy profile any Nash dynamics converges to a Nash equilibrium of smaller or equal social cost.

Theorem 4.8. *For any game $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ in* S-PMC*:*

1. *the price of stability is* 1*, and*
2. *any Nash dynamics converges to a Nash equilibrium in at most* $4^{|\mathcal{P}|}$ *steps.*

Proof. Let $\prec$ denote the standard lexicographic ordering between vectors of equal size. If $\vec{c}$ is a strategy profile for $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ that is not a Nash equilibrium and $\vec{c}'$ is the strategy profile resulting from a profitable deviation of some player i, we show that $\boldsymbol{D}(\vec{c}') \prec \boldsymbol{D}(\vec{c})$ and hence $\mathrm{sc}(\vec{c}') \leq \mathrm{sc}(\vec{c})$. This implies that any Nash dynamics starting from a minimum-cost strategy profile converges to a Nash equilibrium of the same social cost, thus the price of stability is 1.

Since player i profited by deviating, her disutility in the new strategy profile $\vec{c}'$ is reduced by at least 1. Some of the players that overlap with p_i and are colored with c_i may also have their disutilities reduced by exactly 1. The original disutility of any such player p_j must be $f_j(\vec{c}) = f_i(\vec{c})$. The deviation of player i may result in an increase by exactly 1 of the disutility of some players who overlap with p_i and are colored with c_i'. For any player p_k whose disutility is increased, $f_k(\vec{c}) \leq f_i(\vec{c}) - 2$ otherwise p_i would be blocked from switching to p_k's color. The disutilities of all other players remain the same. It is clear now that all players whose disutility changed have a new disutility smaller than $f_i(\vec{c})$. Therefore $\boldsymbol{D}(\vec{c}') \prec \boldsymbol{D}(\vec{c})$.

Regarding the rate of convergence, observe that for any strategy profile $\vec{c}$ the sum of the components of the corresponding disutility vector $\boldsymbol{D}(\vec{c})$ is:

$$\sum_{i=1}^{L} d_i(\vec{c}) = |\mathcal{P}| \quad , \tag{4.16}$$

independent of $\vec{c}$. So, the number of distinct disutility vectors is at most equal to the number of distinct ways in which $|\mathcal{P}|$ indistinguishable balls can be thrown in L bins. This number is known to be

$$\binom{|\mathcal{P}| + L - 1}{|\mathcal{P}|} \leq 2^{|\mathcal{P}|+L-1} < 4^{|\mathcal{P}|} \quad , \tag{4.17}$$

because $L \leq |\mathcal{P}|$. The convergence of any Nash dynamics in at most this many steps follows immediately. □

4.5 Computing Optimal and Approximate Equilibria

Due to Theorem 4.8, computing a Nash equilibrium of minimum social cost is at least as hard as the corresponding optimization problem in which one is given a graph G, a set of simple path $\mathcal{P}$ defined on G, and the number of available colors k and is asked to color all paths in $\mathcal{P}$ so that the maximum fiber multiplicity $\mu_{\max}$ is minimized. As noticed in [62], this problem is NP-hard in general graphs, in fact even in rings and stars. Therefore, it is also NP-hard to compute an optimal Nash equilibrium even in the case of rings and stars. However, we show that there exists an efficient algorithm that computes optimal Nash equilibria for a subclass of S-PMC(TREE). Furthermore, we show that we can use a known algorithm

Algorithm 9 Computing pure Nash equilibria for the class of S-PMC(ROOTED-TREE) games.

Input: an S-PMC(ROOTED-TREE) game $\langle\langle G, \mathcal{P}, k \rangle\rangle$

Find a node r such that each path in $\mathcal{P}$ lies on some path from r to a leaf.
for all edges $e \in E$ in order of non-decreasing distance from r, breaking ties arbitrarily **do**
  for all uncolored paths p that contain edge e **do**
    Pick a color c such that $\mu(e, c)$ is minimum in the current coloring, breaking ties arbitrarily.
    Color p with color c.
  end for
end for

for PATH MULTICOLORING in stars to compute approximate Nash equilibria for S-PMC(STAR) games.

Definition 4.9. *We define* S-PMC(ROOTED-TREE) *to be the subclass of* S-PMC *that contains games $\langle\langle G, \mathcal{P}, k \rangle\rangle$ with the following property:*

> *"G is a tree and there is a node r such that each path in $\mathcal{P}$ lies entirely on some simple path from r to a leaf."*

A similar class of graphs has been defined and studied as an intersection model for "rooted directed edge path graphs" in [59].

We will say that a path in a tree rooted at r *starts* on edge e, if e is the edge of the path that lies closest to r. Algorithm 9 is a polynomial-time algorithm that computes optimal Nash equilibria for S-PMC(ROOTED-TREE) games. We proceed to prove its correctness.

Lemma 4.10. *Given an* S-PMC(ROOTED-TREE) *game as input, Algorithm 9 computes a pure Nash equilibrium.*

Proof. Let $\langle\langle G, \mathcal{P}, k \rangle\rangle$ be the input game, where $G = (V, E)$ is a tree, and r be the root of the tree. Also, let $e_1, \ldots, e_{|E|}$ be the order in which Algorithm 9 considers the edges of E. We will prove that the outer loop of Algorithm 9 maintains the following invariant:

> "after the j-th iteration, all those paths in $\mathcal{P}$ that have been colored are in Nash equilibrium."

In the rest of the proof, we will denote by $\mu_j(e, c)$ the multiplicity of color c on edge e after the j-th iteration of the outer loop of Algorithm 9. Likewise, we will denote by $\mu_j(p, c)$ the maximum multiplicity of color c over all edges of path p after the j-th iteration, i.e.,

$$\mu_j(p, c) = \max_{e \in p} \mu_j(e, c) \ . \tag{4.18}$$

For $j = 1$, the multiplicity of any color on edge e_1 is either $\left\lfloor \frac{L(e_1)}{k} \right\rfloor$ or $\left\lceil \frac{L(e_1)}{k} \right\rceil$. These numbers differ by at most 1, so the paths that start on edge e_1 are in Nash equilibrium after the first iteration.

For $j > 1$, assume that the invariant holds after the $(j - 1)$-st iteration. Note that all paths not using e_j are not affected by the j-th iteration. We shall now prove that no path that uses edge e_j has incentive to change color after the j-th iteration.

First, let p be a path that had not been considered before, i.e., p starts on edge e_j. Let a_i be its color. Consider now the path, say p', that was last colored with a_i during the j-th iteration. At the moment p' was colored, a_i must have been a minimum multiplicity color, therefore e_j is an a-blocking edge for p for all $a \neq a_i$.

Now, let p be a path that uses edge e_j and is already colored at the beginning of the j-th iteration with color a_i. We distinguish between two cases:

- $\mu_j(p, a_i) = \mu_{j-1}(p, a_i)$: since the paths starting on edges $e_1, \ldots, e_{j-1}$ were in Nash equilibrium after the $(j-1)$-st iteration, for each color $a \neq a_i$ there was an a-blocking edge along path p. These edges are still a-blocking for p, because the maximum multiplicity of a_i along p has not changed. Therefore, p has no incentive to change color.

- $\mu_j(p, a_i) > \mu_{j-1}(p, a_i)$: this increase in the maximum multiplicity of a_i along p must be due to one or more new paths starting on edge e_j being colored with a_i. Let p' be the last of these paths that was colored with a_i. At the moment p' was colored, a_i must have been a minimum-multiplicity color. Therefore, for any color a we have that

$$\mu_j(e_j, a) \geq \mu_j(e_j, a_i) - 1 \ . \tag{4.19}$$

  But $\mu_j(e_j, a_i) = \mu_j(p, a_i)$, because the maximum multiplicity of a_i along p has just increased at edge e_j. This implies that edge e_j is an a-blocking edge for p for any $a \neq a_i$, therefore p has no incentive to change color.

□

Theorem 4.11. *Given an* S-PMC(ROOTED-TREE) *game as input, Algorithm 9 computes an optimal Nash equilibrium of minimum social cost.*

Proof. By Lemma 4.10, the output of the algorithm is a Nash equilibrium. Let μ_j be the maximum multiplicity of any color after the j-th iteration of the outer loop of Algorithm 9. Using the notation of the proof of Lemma 4.10,

$$\mu_j = \max_{1 \le i \le j} \max_{a \in W} \mu_j(e_i, a) \ . \tag{4.20}$$

It is implicit above that for any $j \geq 1$, if $\mu_j > \mu_{j-1}$ then μ_j changed due to the coloring of certain paths on e_j, hence

$$\mu_j = \mu(e_j, a_i) = \left\lceil \frac{L(e_j)}{k} \right\rceil \ , \tag{4.21}$$

for some color a_i. If the maximum multiplicity $\mu_{\max}$ was last increased while processing edge e, then by the previous remark the algorithm produces a coloring with cost

$$\mu_{\max} = \left\lceil \frac{L(e)}{k} \right\rceil \leq \left\lceil \frac{L}{k} \right\rceil \ . \tag{4.22}$$

Since $\left\lceil \frac{L}{k} \right\rceil$ is a lower bound for μ_{OPT}, it turns out that

$$\mu_{\max} = \mu_{OPT} = \left\lceil \frac{L}{k} \right\rceil \ . \tag{4.23}$$

Therefore, edge e must be a maximum-load edge and the strategy profile output by the algorithm must have social cost equal to $\left\lceil \frac{L}{k} \right\rceil$. $\square$

Theorem 4.12. *There is a polynomial-time algorithm that computes a $\frac{1}{2}$-approximate Nash equilibrium for any* S-PMC(STAR) *game.*

Proof. Let $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ be a game in S-PMC(STAR). We use the polynomial-time approximation algorithm presented in [62] for the PATH MULTICOLORING problem in stars, which returns a coloring of the paths in $\mathcal{P}$ with the following property: for any edge e and any color c there exist integers a, b such that

$$\left\lceil \frac{a}{k} \right\rceil + \left\lceil \frac{b}{k} \right\rceil - 2 \leq \mu(e, c) \leq \left\lceil \frac{a}{k} \right\rceil + \left\lceil \frac{b}{k} \right\rceil \ , \tag{4.24}$$

and $a + b = L(e)$. This implies that if $\vec{c}$ is the strategy profile returned by the algorithm, then any player i who deviates resulting in a new strategy profile $\vec{c}'$ reduces her cost by at most 1. Therefore,

$$f_i(\vec{c}') = f_i(\vec{c}) - 1 = \left(1 - \frac{1}{f_i(\vec{c})}\right) \cdot f_i(\vec{c}) \ . \tag{4.25}$$

In the worst case $f_i(\vec{c}) = 2$, hence $\vec{c}$ is an $\frac{1}{2}$-approximate Nash equilibrium. $\square$

4.6 Tight Upper Bounds on the Price of Anarchy

In this section we provide two upper bounds on the price of anarchy of any S-PMC game $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ and we show that both of them are tight. The first bound is determined by a property of the network, namely the number of available wavelengths. The second bound is more subtle, as it depends on the length of paths with the highest disutility in worst-case Nash equilibria. We prove that these bounds are tight even for the class S-PMC(ROOTED-TREE), and asymptotically tight for the class S-PMC(ROOTED-TREE: $\Delta = 3$), i.e., the subclass of S-PMC(ROOTED-TREE) that contains games defined on graphs with maximum degree 3.

Lemma 4.13. *The price of anarchy of any* S-PMC *game* $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ *is at most* k*.*

Proof. Let $\vec{c}$ be a worst-case Nash equilibrium of $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$, hence $\mathrm{sc}(\vec{c}) = \hat{\mu}$. Clearly, $\hat{\mu} \le L$ and since the minimum social cost over all strategy profiles is $\mu_{\mathrm{OPT}} \ge \left\lceil \frac{L}{k} \right\rceil$, it turns out that $\mu_{\mathrm{OPT}} \ge \frac{\hat{\mu}}{k}$. This implies that $\frac{\hat{\mu}}{\mu_{\mathrm{OPT}}} \le k$. □

The next lemma is an arithmetical lemma that will be used in the proof of Lemma 4.15.

Lemma 4.14. *Let*

$$f(\hat{\mu}, w, z) = \frac{\hat{\mu}}{\left\lceil \frac{1 + \lceil \frac{w}{z} \rceil (\hat{\mu} - 1)}{w} \right\rceil} \ . \tag{4.26}$$

It holds that $f(\hat{\mu}, w, z) \le z$*, for all* $\hat{\mu} \ge 2$*,* $w \ge 1$*, and* $z \ge 2$*.*

Proof. We distinguish the following four cases:

- $w = \kappa z$ and $\hat{\mu} = \lambda z$, for some $\kappa \ge 1$, $\lambda \ge 1$:

$$f(\lambda z, \kappa z, z) = \frac{\lambda z}{\left\lceil \frac{1 + \kappa(\lambda z - 1)}{\kappa z} \right\rceil} = \frac{\lambda z}{\left\lceil \lambda - \frac{\kappa - 1}{\kappa z} \right\rceil} = z \ . \tag{4.27}$$

  The last equality holds because $\lambda - 1 < \lambda - \frac{\kappa-1}{\kappa z} \le \lambda$.

- $w = \kappa z$ and $\hat{\mu} = \lambda z + \chi$, for some $\kappa \ge 1$, $\lambda \ge 0$, $1 \le \chi \le z - 1$:

$$f(\lambda z + \chi, \kappa z, z) = \frac{\lambda z + \chi}{\left\lceil \frac{1 + \kappa(\lambda z + \chi - 1)}{\kappa z} \right\rceil} = \frac{\lambda z + \chi}{\left\lceil \lambda + \frac{\kappa \chi - \kappa + 1}{\kappa z} \right\rceil} \ . \tag{4.28}$$

  Because $\chi \ge 1$ and $\chi \le z - 1$, we get that

$$f(\lambda z + \chi, \kappa z, z) \le \frac{\lambda z + z - 1}{\left\lceil \lambda + \frac{\kappa - \kappa + 1}{\kappa z} \right\rceil} = \frac{\lambda z + z - 1}{\lambda + 1} < z \ . \tag{4.29}$$

- $w = \kappa z + \varphi$ and $\hat{\mu} = \daleth z$, for some $\kappa \geq 0$, $1 \leq \varphi \leq z-1$, $\daleth \geq 1$:

$$f(\daleth z, \kappa z + \varphi, z) = \frac{\daleth z}{\left\lceil \frac{1+(\kappa+1)(\daleth z-1)}{\kappa z+\varphi} \right\rceil} = \frac{\daleth z}{\left\lceil \daleth + \frac{\daleth(z-\varphi)}{\kappa z+\varphi} - \frac{\kappa}{\kappa z+\varphi} \right\rceil} \ , \tag{4.30}$$

where the first equality follows from the fact that $\left\lceil \frac{w}{z} \right\rceil = \left\lceil \kappa + \frac{\varphi}{z} \right\rceil = \kappa+1$. Because $\frac{\daleth(z-\varphi)}{\kappa z+\varphi} > 0$ we get that

$$f(\daleth z, \kappa z + \varphi, z) \leq \frac{\daleth z}{\left\lceil \daleth - \frac{\kappa}{\kappa z+\varphi} \right\rceil} \ , \tag{4.31}$$

and because $\frac{\kappa}{\kappa z+\varphi} < 1$ we have $\left\lceil \daleth - \frac{\kappa}{\kappa z+\varphi} \right\rceil = \daleth$ and therefore

$$f(\daleth z, \kappa z + \varphi, z) \leq \frac{\daleth z}{\daleth} = z \ . \tag{4.32}$$

- $w = \kappa z + \varphi$ and $\hat{\mu} = \daleth z + \chi$, for some $\kappa \geq 0$, $1 \leq \varphi \leq z-1$, $\daleth \geq 0$, $1 \leq \chi \leq z-1$:

$$f(\daleth z + \chi, \kappa z + \varphi, z) = \frac{\daleth z + \chi}{\left\lceil \frac{1+(\kappa+1)(\daleth z+\chi-1)}{\kappa z+\varphi} \right\rceil} = \frac{\daleth z + \chi}{\left\lceil \daleth + \frac{\daleth(z-\varphi)+\kappa(\chi-1)+\chi}{\kappa z+\varphi} \right\rceil} \ . \tag{4.33}$$

Because $z > \varphi$, $\chi \geq 1$ and $\chi \leq z-1$, we get that

$$f(\daleth z + \chi, \kappa z + \varphi, z) \leq \frac{\daleth z + z - 1}{\daleth + 1} < z \ . \tag{4.34}$$

□

Lemma 4.15. *For any worst-case Nash equilibrium $\vec{c}$ of an* S-PMC *game $\langle\langle G, \mathcal{P}, k \rangle\rangle$ and for any $p_i \in \mathcal{P}$ with $f_i(\vec{c}) = \mathrm{sc}(\vec{c}) = \hat{\mu}$, the price of anarchy of $\langle\langle G, \mathcal{P}, k \rangle\rangle$ is at most equal to the length of p_i.*

Proof. Let $\tilde{e}$ be an edge of p_i where the color c_i chosen by p_i appears with maximum multiplicity $\hat{\mu}$: $\mu(\tilde{e}, c_i) = \hat{\mu}$. Let z denote the length of path p_i and let $e_1, \ldots, e_{z-1}$ be the edges that p_i uses, apart from $\tilde{e}$. For $1 \leq j \leq z-1$, let x_j be the number of colors that are blocked for p_i on e_j and let y be the number of colors that are blocked for p_i on $\tilde{e}$. Since $\vec{c}$ is a Nash equilibrium, it must be that

$$x_1 + \ldots + x_{z-1} + y \geq k - 1 \ . \tag{4.35}$$

If it is the case that $z = 1$, i.e. p_i uses only edge $\tilde{e}$, then $\tilde{e}$ must block all colors for p_i except c_i. This implies that the load of edge $\tilde{e}$ is:

$$L(\tilde{e}) \geq \hat{\mu} + (k-1)(\hat{\mu}-1) = k\hat{\mu} - k + 1 \ . \tag{4.36}$$

Therefore, the minimum social cost over all strategy profiles satisfies:

$$\mu_{\mathrm{OPT}} \geq \left\lceil \frac{L}{k} \right\rceil \geq \left\lceil \frac{L(\tilde{e})}{k} \right\rceil \geq \left\lceil \hat{\mu} - \frac{k-1}{k} \right\rceil = \hat{\mu} \ . \tag{4.37}$$

We conclude that the price of anarchy in this case is equal to 1.

Now, assume that $z \geq 2$. We will prove that

$$L \geq 1 + \left\lceil \frac{k}{z} \right\rceil (\hat{\mu} - 1) \ . \tag{4.38}$$

First, observe that

$$L(\tilde{e}) \geq \hat{\mu} + y(\hat{\mu} - 1) \tag{4.39}$$

and, for $1 \leq j \leq z-1$,

$$L(e_j) \geq 1 + x_j(\hat{\mu} - 1) \ . \tag{4.40}$$

If $y \geq \left\lceil \frac{k}{z} \right\rceil - 1$, then

$$L \geq L(\tilde{e}) \geq \hat{\mu} + \left(\left\lceil \frac{w}{z} \right\rceil - 1 \right)(\hat{\mu} - 1) = 1 + \left\lceil \frac{w}{z} \right\rceil (\hat{\mu} - 1) \ . \tag{4.41}$$

If, on the other hand, $y < \left\lceil \frac{k}{z} \right\rceil - 1$, then Equation 4.35 gives

$$x_1 + \ldots + x_{z-1} \geq k - 1 - y \geq k - \left\lceil \frac{k}{z} \right\rceil + 1 \ . \tag{4.42}$$

This implies that there is some x_s such that

$$x_s \geq \frac{k - \left\lceil \frac{k}{z} \right\rceil + 1}{z-1} > \frac{k - \frac{k}{z} - 1 + 1}{z-1} = \frac{k}{z} \ . \tag{4.43}$$

Since x_s is an integer, it must be that $x_s \geq \left\lceil \frac{k}{z} \right\rceil$. Therefore,

$$L \geq L(e_s) \geq 1 + \left\lceil \frac{k}{z} \right\rceil (\hat{\mu} - 1) \ . \tag{4.44}$$

We conclude that Equation 4.38 holds in any case. So, the price of anarchy is bounded as follows:

$$\mathrm{PoA}\left(\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle\right) = \frac{\hat{\mu}}{\mu_{\mathrm{OPT}}} \leq \frac{\hat{\mu}}{\left\lceil \frac{L}{k} \right\rceil} \leq \frac{\hat{\mu}}{\left\lceil \frac{1 + \left\lceil \frac{k}{z} \right\rceil (\hat{\mu} - 1)}{k} \right\rceil} \leq z \ . \tag{4.45}$$

The last inequality holds by Lemma 4.14. $\square$

As an immediate corollary of Lemma 4.15, we derive the following upper bound on the price of anarchy:

Corollary 4.16. *The price of anarchy of any* S-PMC *game* $\langle\langle G, \mathcal{P}, k\rangle\rangle$ *is upper-bounded as follows:*

$$\mathrm{PoA}\left(\langle\langle G, \mathcal{P}, k\rangle\rangle\right) \leq \min_{\vec{c}:\mathrm{sc}(\vec{c})=\hat{\mu}} \min_{i:f_i(\vec{c})=\hat{\mu}} |p_i| \ . \tag{4.46}$$

Lemma 4.17. *The upper bounds of Lemma 4.13 and Corollary 4.16 are tight even for the class of* S-PMC(ROOTED-TREE) *games.*

Proof. We first define a recursive construction of an S-PMC game and a Nash equilibrium for this game. The construction is illustrated in Figure 4.1.

For any $z \geq 1$ and $\hat{n} \geq 1$, let $\mathcal{A}_z(\hat{n})$ be the following S-PMC game with z available colors: there are $\hat{n}$ paths of color a_1 and length z, starting at the *root node* u_0, which branch out into $\hat{n}$ branches, one on each branch. Let us call these the *primary* paths for $\mathcal{A}_z(\hat{n})$. On any of the $z-1$ edges of each such branch, one color is blocked for the primary path. The $\hat{n}-1$ blocking paths of each edge branch out into an $\mathcal{A}_z(\hat{n}-1)$ game. They become primary paths for this copy of $\mathcal{A}_z(\hat{n}-1)$. The root node for the j-th recursive copy of $\mathcal{A}_z(\hat{n}-1)$ on the i-th branch is node $u_{i,j}$ (node $u_{i,1}$ is common for all branches). The base case of this recursive construction is $\mathcal{A}_z(0)$, which is a degenerate game with no paths and no available colors, defined on a graph consisting of a single node. We have included the explicit construction for $z = \hat{n} = 3$ in Figure 4.2.

Claim 4.18. *For any* $z \geq 1$*, the construction* $\mathcal{A}_z(z)$ *is an* S-PMC(ROOTED-TREE) *game in Nash equilibrium, in which all of the following are equal to* z*: the number of available colors* k*, the maximum load* L*, the maximum color multiplicity* $\mu_{\max}$*, and all path lengths.*

Proof (of Claim). It is straightforward to verify that $\mathcal{A}_z(z)$ belongs to the class S-PMC(ROOTED-TREE); the root node is the root node u_0 of the first level of the recursive construction. The game is in Nash equilibrium by construction, since every path contains one blocking edge for every color other than its own. The number of available colors is equal to z by definition. The maximum multiplicity $\mu_{\max} = z$ appears on the edge incident to the root node of $\mathcal{A}_z(z)$. The maximum load $L = z$ appears on all the edges of the first level of the construction. Finally, all path lengths are equal to z by construction. The claim is proved. □

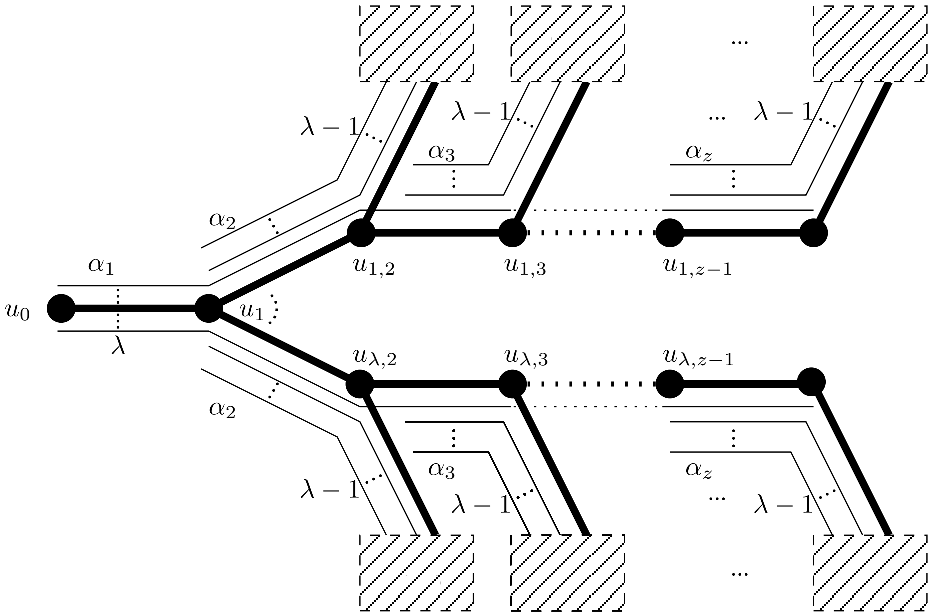

Figure 4.1: The construction $\mathcal{A}_z(\hbar)$ for the proof of Lemma 4.17. The thick lines represent the edges of the underlying graph, and the thin lines represent the paths defined on the graph. The color and multiplicity of each group of paths is written next to that group. Each shaded box represents a recursive copy of $\mathcal{A}_z(\hbar - 1)$.

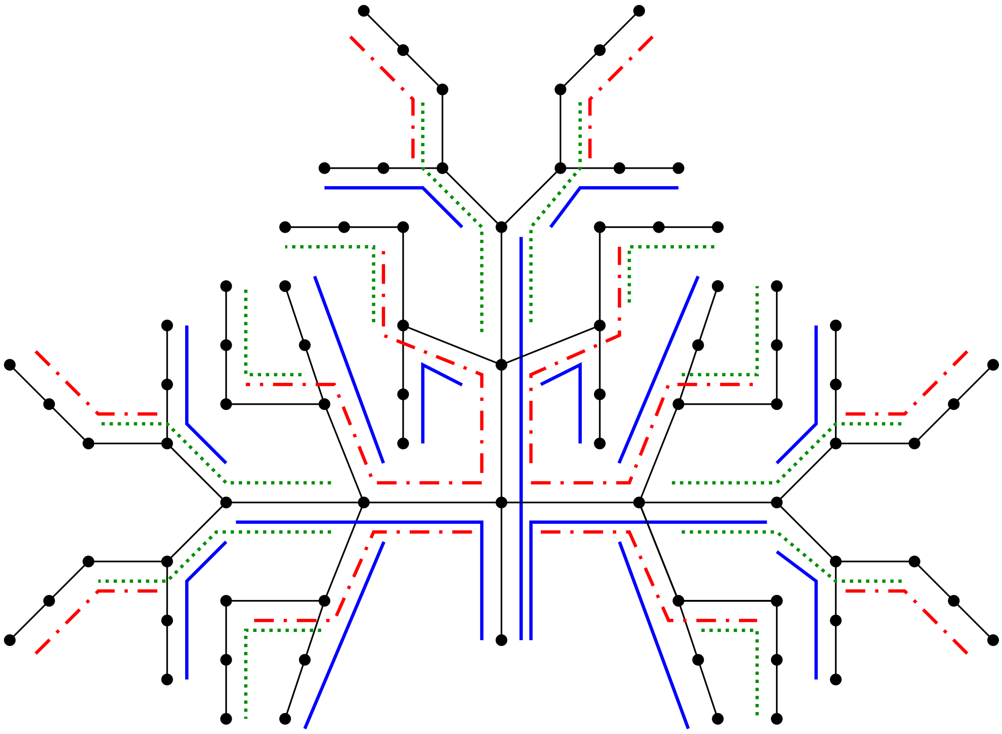

Figure 4.2: The construction $\mathcal{A}_3(3)$, as described in Lemma 4.17. Different colors are shown by different line styles. Solid black lines represent the edges of the underlying graph.

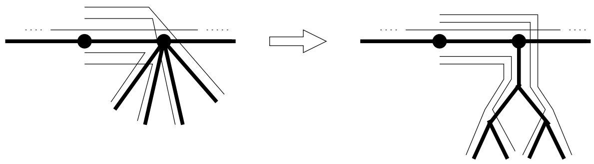

Figure 4.3: Alternate branching in the construction of Lemma 4.17 in order to achieve an asymptotically tight lower bound for the price of anarchy on graphs with maximum degree 3.

By Theorem 4.11, the optimal strategy profile for $\mathcal{A}_z(z)$ has social cost $\mu_{\mathrm{OPT}} = \left\lceil \frac{L}{k} \right\rceil = 1$. Therefore, the ratio $\frac{\mu_{\max}}{\mu_{\mathrm{OPT}}}$ is equal to z for this Nash equilibrium, hence the price of anarchy is at least z. □

Lemma 4.19. *The upper bounds of Lemma 4.13 and Corollary 4.16 are asymptotically tight even for the class of* S-PMC(ROOTED-TREE) *games with maximum degree* 3.

Proof. The construction presented in Lemma 4.17 can be modified so that the maximum degree of the resulting tree is 3, with only a logarithmic increase in the length of the paths: whenever a multitude of d paths that use the same edge spread out into d branches so that only one of them lies on each branch, let the branching not be effected immediately, but let instead the paths carry on for one additional edge in the direction in which they are headed, then split them onto 2 new edges (half of the paths on each edge), then split the paths of each edge onto 2 new edges, and so on until each path is singled out. An example of this modification for $d = 4$ is illustrated in Figure 4.3.

It is easy to see that this process results in a tree with maximum degree 3 and increases the length of the paths by at most $\lceil \log d \rceil$. If we take care to add edges where needed so that all paths have the same length and carry out this process for the whole tree, then we result in an S-PMC(ROOTED-TREE) game and a Nash equilibrium thereof with the same properties as the construction in Lemma 4.17, except that the maximum degree is 3 and the length of all paths is $z' = z + O(\log z)$. It turns out, then, that $\mathrm{PoA} \geq z = z' - O(\log z) = z' - o(z')$. □

We summarize the results of Lemmata 4.13, 4.15, 4.17, and 4.19 in the following theorem:

Theorem 4.20. *The price of anarchy of any* S-PMC *game* $\langle\langle G, \mathcal{P}, k \rangle\rangle$ *is upper-bounded both by* k *and by*

$$\min_{\vec{c}:\mathrm{sc}(\vec{c})=\hat{\mu}} \; \min_{i:f_i(\vec{c})=\hat{\mu}} |p_i| \quad . \tag{4.47}$$

These bounds are tight for the class S-PMC(ROOTED-TREE) *and asymptotically tight for the class* S-PMC(ROOTED-TREE: $\Delta = 3$).

Theorem 4.21. *The price of anarchy of the class* S-PMC(STAR) *is* 2.

Proof. Lemma 4.15 implies an upper bound of 2 on the price of anarchy, since the length of any simple path defined on a star cannot be larger than 2.

For the lower bound, we can easily modify the construction that appears in Lemma 4.17 to yield a family of S-PMC(STAR) games with price of anarchy 2. More specifically, observe that in any game $\mathcal{A}_2(\hbar)$ we have only players (paths) of length 2. Such a game can be embedded in a star with exactly the same number of edges as follows: fix an isomorphism φ between the edges of the tree and the edges of the star, and for every path $p = \{e, e'\}$ defined on the tree, define a path $\tilde{p} = \{\varphi(e), \varphi(e')\}$ of the same color on the star. It is clear that the paths we just defined on the star overlap with each other in exactly the same way as the original paths overlapped on the tree. Therefore, the game on the star is in Nash equilibrium with $\mu_{\max} = \hbar$, whereas the optimal solution has maximum color multiplicity $\mu_{\mathrm{OPT}} = \frac{\hbar}{2}$. □

4.7 The Price of Anarchy on Graphs with Maximum Degree 2

In this section we study the price of anarchy of path multicoloring games on chains and rings, and we prove a constant upper bound for a broad class of S-PMC(RING) games with $L = \Omega(k^2)$. Notice that this class essentially encompasses all S-PMC(RING) games of practical importance, as the number of wavelengths is limited in practice due to technological constraints, whereas L can be arbitrarily large depending on network traffic. For the sake of completeness, we show that the price of anarchy may become unbounded if the network designer opts to provide ample wavelengths to the users, i.e., when $L = o(k^2)$.

We begin by showing a more involved necessary condition for Nash equilibria of S-PMC(RING) games than the one we have already seen in Property 4.6. Let $\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle$ be an S-PMC(RING) game. Given a coloring $\vec{c} = (c_1, \ldots, c_{|\mathcal{P}|})$, let $P_{e,a_i}(\vec{c}) \subseteq \mathcal{P}$ denote the set of paths colored with color a_i that use edge $e \in E$. We have, by definition, $\left|P_{e,a_i}(\vec{c})\right| = \mu(e, a_i)$. For the sake of simplicity, in the rest of the section we will write P_{e,a_i} instead of $P_{e,a_i}(\vec{c})$. Furthermore, let $[e, e']$ denote the clockwise arc starting at edge e and ending at edge e'.

Lemma 4.22 (Structural property of S-PMC(RING) Nash equilibria)**.** *Given an* S-PMC(RING) *game and a coloring $\bar{c}$ thereof which is a Nash equilibrium, for every edge e and color a_i there is an edge-simple arc $[e_l, e_r]$ with the following properties:*

1. *for every color $a_j \neq a_i$, arc $[e_l, e_r]$ contains an edge which is an a_j-blocking edge for at least half of the paths in P_{e,a_i}, and*

2. *for every edge e' of the arc $[e_l, e_r]$ it holds that*

$$\left|P_{e',a_i} \cap P_{e,a_i}\right| \geq \left\lceil \frac{\left|P_{e,a_i}\right|}{2} \right\rceil . \tag{4.48}$$

Proof. Since the game is in Nash equilibrium, by Property 4.6 every path $p \in P_{e,a_i}$ must have at least one a_j-blocking edge, for every color $a_j \neq a_i$. For a fixed color $a_j \neq a_i$, consider the two a_j-blocking edges for some path in P_{e,a_i} that are closest to edge e clockwise and counter-clockwise. It is not hard to see that for at least one of these two edges, call it $b(a_j)$, the following property holds: the arc $[e, b(a_j)]$ or the arc $[b(a_j), e]$ is contained in at least $\left\lceil \frac{|P_{e,a_i}|}{2} \right\rceil$ of the paths in P_{e,a_i}. In case that there is only one a_j-blocking edge for all paths in P_{e,a_i}, then the property holds *a fortiori* for this edge.

For every color a_j we pick one such edge $b(a_j)$. If the above property holds for arc $[e, b(a_j)]$, we add $b(a_j)$ to set B^+, otherwise we add it to set B^-. We now claim that a clockwise traversal of the ring starting at edge e will first encounter all edges of B^+ and then all edges of B^-. Indeed, if one edge b^- of B^- lies before one edge b^+ of B^+ on this clockwise traversal, this would imply that b^- is traversed by the $\left\lceil \frac{|P_{e,a_i}|}{2} \right\rceil$ paths that contain the arc $[e, b^+]$ and thus b^- should also belong to B^+.

The above discussion implies that if we define e_r to be the last edge of B^+ and e_l to be the first edge of B^- encountered in this clockwise traversal, then the edge-simple arc $[e_l, e_r]$ satisfies the conditions of the Lemma. □

Definition 4.23. *We define $[e_l, e_r]_{P_{e,a_i}}$ to be the arc that is obtained by applying Lemma 4.22 for path set P_{e,a_i}. We shall also denote the extreme edges of the arc by $e_l\,(P_{e,a_i})$ and $e_r\,(P_{e,a_i})$.*

4.7.1 A Constant Bound on the Price of Anarchy for Small Number of Wavelengths

In this section we prove a constant upper bound on the price of anarchy of S-PMC(RING) games with $L = \Omega(k^2)$; denote this class by S-PMC(RING:

$L = \Omega(k^2)$). This provides an upper bound on the price of anarchy of any S-PMC(CHAIN: $L = \Omega(k^2)$) game as well, since every game defined on a chain can be trivially embedded in a ring topology.

We first employ the structural property of S-PMC(RING) Nash equilibria (Lemma 4.22) in order to establish the existence of a heavily loaded edge in S-PMC(RING) games with $\hat{\mu} \geq k$. To that end, we make use of a recursive argument. In particular, we define $\tilde{e}$ to be the edge where the maximum color multiplicity occurs in a worst-case Nash equilibrium, and without loss of generality we consider this color to be a_1. We then observe that all a_j-blocking paths for paths in $P_{\tilde{e},a_1}$ must also be in equilibrium, hence recursively there exist a_s-blocking paths for them, for all $a_s \neq a_j$ and so on. In order to establish the desired load we prove a bound on the number of these paths that use the same edge.

Lemma 4.24. *In every* S-PMC(RING) *game* $\langle\langle G, \mathcal{P}, k\rangle\rangle$ *with* $\hat{\mu} \geq k$ *there is an edge with load at least* $\frac{\hat{\mu}k}{4}$.

Proof. Consider a coloring $\vec{c}$ which is a worst-case Nash equilibrium for $\langle\langle G, \mathcal{P}, k\rangle\rangle$. We define P_1 to be the set of paths $P_{\tilde{e},a_1}$ which induce the social cost $\hat{\mu}$. For $i \geq 2$ we define P_i to be the set of a_j-blocking paths for the path set P_{i-1}, for some color a_j not appearing in any of the path sets P_s, $s < i$, with the following property:

$$[e_l, e_r]_{P_i} \subseteq [e_l, e_r]_{P_{i-1}} \quad , \tag{4.49}$$

if such a path set exists. If more than one path sets with the desired property exist, we arbitrarily pick one of them.

Let e_i be the a_j-blocking edge for P_{i-1}. Based on the definition of P_i as a set of blocking paths for path set P_{i-1} we can easily show that $\mu(e_i, a_j) \geq \hat{\mu} - i + 1$. The application of Lemma 4.22 for color a_j and edge e_i yields that for every edge $e \in [e_l, e_r]_{P_i}$ we have that

$$\mu(e, a_j) \geq \frac{\hat{\mu} - i + 1}{2} \quad . \tag{4.50}$$

Furthermore, since Equation 4.49 holds for all $s \leq i$, the load of each edge $e \in [e_l, e_r]_{P_i}$ is at least $\sum_{a_j} \mu(e, a_j)$, where a_j now ranges over the colors of all path sets P_s, $s \leq i$. Hence, for every edge $e \in [e_l, e_r]_{P_i}$ we have that

$$L(e) \geq \sum_{a_j} \mu(e, a_j) \geq \sum_{s=1}^{i} \frac{\hat{\mu} - s + 1}{2} \quad . \tag{4.51}$$

Let now n be the first integer for which no such path set P_n exists (see Figure 4.4) and consider the path set P_{n-1}. Since we are in Nash

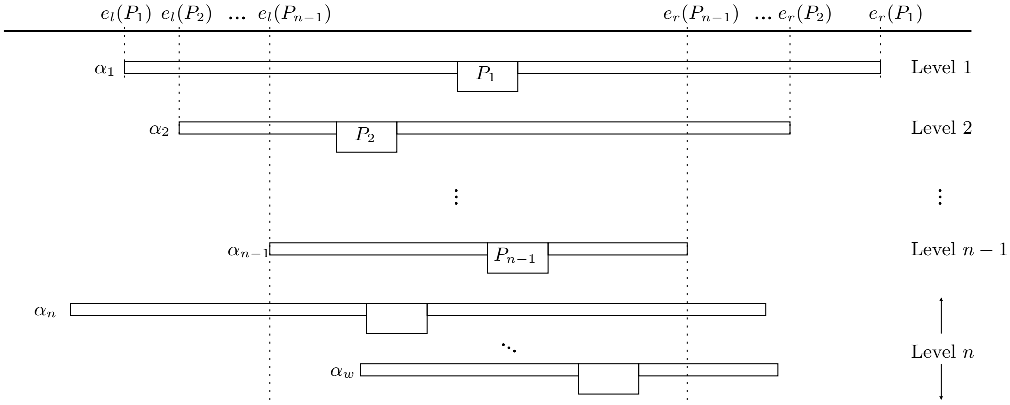

Figure 4.4: The path structure implied in the proof of Lemma 4.24. For the sake of simplicity, paths in P_i are assumed to be colored with α_i, for $i < n$.

equilibrium we know that there exist α-blocking edges for paths in P_{n-1}, for every color α. We restrict our attention to the $k-n+1$ colors which have not yet appeared in any P_s, for $s \leq n-1$. Let α_j be one of these colors. Consider now an α_j-blocking edge e_n such that $e_n \in [e_l, e_r]_{P_{n-1}}$ (by Lemma 4.22 such an edge must exist). We now have that, at least half of the α_j-blocking paths in P_{e_n,α_j}, i.e. at least $\frac{\hat{\mu}-n+1}{2}$ paths, extend beyond one of the extreme edges $e_l(P_{n-1})$ and $e_r(P_{n-1})$ of the arc $[e_l, e_r]_{P_{n-1}}$ (otherwise we would have picked P_{e_n,α_j} to be P_n). This means that for at least half of these $k-n+1$ blocking path sets, their paths leave the arc from the same edge, incurring on it an additional load of $\frac{k-n+1}{2} \cdot \frac{\hat{\mu}-n+1}{2}$.

Thus, the total load of this edge is at least

$$\sum_{i=1}^{n-1} \frac{\hat{\mu}-i+1}{2} + \frac{k-n+1}{2} \cdot \frac{\hat{\mu}-n+1}{2} = \frac{\hat{\mu}k}{4} + (n-1) \cdot \frac{\hat{\mu}-k+1}{4} \quad . \tag{4.52}$$

Since $\hat{\mu} \geq k$ the above sum is at least $\frac{\hat{\mu}k}{4}$. □

We are now ready to prove a constant upper bound on the price of anarchy of games in S-PMC(RING: $L = \Omega(k^2)$).

Theorem 4.25. *The price of anarchy of any game in the class* S-PMC(RING: $L = \Omega(k^2)$) *is bounded by a constant.*

Proof. Let $\langle\langle G, \mathcal{P}, k\rangle\rangle$ be a game in S-PMC(RING: $L = \Omega(k^2)$). We distinguish between two cases:

- If $\hat{\mu} \geq k$, then by Lemma 4.24 we get $L \geq \frac{\hat{\mu}k}{4}$. This implies that

$$\frac{L}{k} \geq \frac{\hat{\mu}}{4} \Rightarrow \mu_{\text{OPT}} \geq \frac{\hat{\mu}}{4} \Rightarrow \text{PoA}\left(\langle\langle G, \mathcal{P}, k\rangle\rangle\right) \leq 4 \quad . \tag{4.53}$$

- If $\hat{\mu} < k$, then we can bound the price of anarchy as follows:

$$\mathrm{PoA}\left(\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle\right) = \frac{\hat{\mu}}{\mu_{\mathrm{OPT}}} \leq \frac{\hat{\mu}k}{L} < \frac{k^2}{L} \ , \tag{4.54}$$

  where we used successively the facts that $\mu_{\mathrm{OPT}} \geq \frac{L}{k}$ and $\hat{\mu} < k$. The last inequality, combined with the fact that $L = \Omega(k^2)$, implies $\mathrm{PoA}\left(\langle\!\langle G, \mathcal{P}, k \rangle\!\rangle\right) = \mathcal{O}(1)$.

□

4.7.2 Unbounded Price of Anarchy for Large Number of Wavelengths

In this section we show that for any fixed ε in the range $0 < \varepsilon < 1$, there exists an infinite family of S-PMC(CHAIN: $L = \Theta(k^{2-\varepsilon})$) games whose price of anarchy is $\Omega(k^{\frac{\varepsilon}{2}})$. This implies that the price of anarchy can get arbitrarily large when the number of available colors increases, therefore the price of anarchy is unbounded for the classes S-PMC(CHAIN: $L = o(k^2)$) and S-PMC(RING: $L = o(k^2)$).

Theorem 4.26. *For any fixed ε, $0 < \varepsilon < 1$, there exists an infinite family of games in* S-PMC(CHAIN: $L = \Theta(k^{2-\varepsilon})$) *with price of anarchy* $\Omega(k^{\frac{\varepsilon}{2}})$.

Proof. We first observe that for any fixed $\varepsilon > 0$, given an integer $\rho \geq 4$ we can construct an S-PMC(CHAIN: $L = \Theta(k^{2-\varepsilon})$) game and a strategy profile thereof that is a Nash equilibrium, with the following properties:

- the number of available colors is $k = \left\lceil \rho^{1+\frac{\varepsilon}{2-\varepsilon}} \right\rceil$,
- the maximum load is $L = \Theta(\rho^2)$, and
- the maximum multiplicity of any color is $\mu_{\max} = \rho$.

Construction For given values of the parameters ε and ρ, we describe the construction of an S-PMC(CHAIN) game, using the path set $P(A, a_i)$ illustrated in Figure 4.5 as a building block. In what follows we describe the structure of $P(A, a_i)$ along with a coloring of its paths that we will prove to be a Nash equilibrium.

We define $P(A, a_i)$, with $A \subseteq W = \{a_1, \ldots, a_k\}$ and $a_i \in W \setminus A$, to be a path set consisting of:

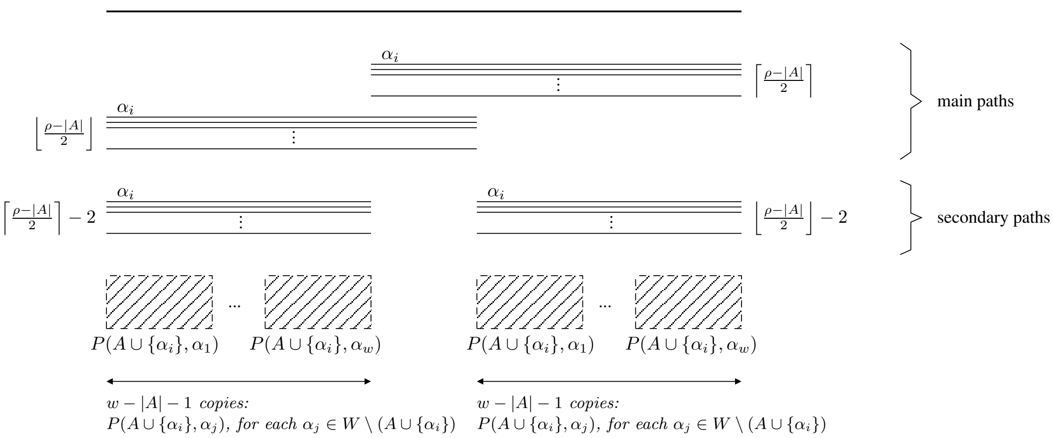

Figure 4.5: Recursive construction of path set $P(A, a_i)$ for $a_i \in W \setminus A$.

- the *main paths*: these are the $\rho - |A|$ paths of color a_i, arranged as shown in the top part of Figure 4.5, overlapping all together on some edge (henceforth called the *central* edge for $P(A, a_i)$) with half of them extending to the left of this edge and the other half extending to the right of it,

- the *secondary paths*: these are the $\rho - |A| - 4$ paths of color a_i, placed below the main paths as shown in Figure 4.5, and

- the copies of $P\left(A \cup \{a_i\}, a_j\right)$, for every $a_j \in W \setminus (A \cup \{a_i\})$, one copy of $P\left(A \cup \{a_i\}, a_j\right)$ to the left of the central edge of $P(A, a_i)$, and one copy to the right. We say that $P(A, a_i)$ *contains* each of the copies $P\left(A \cup \{a_i\}, a_j\right)$ and we write $P(A, a_i) \succ P\left(A \cup \{a_i\}, a_j\right)$. The reflexive and transitive closure of $\succ$ is denoted by $\succ^*$.

We now claim that the S-PMC(Chain) game $\mathcal{B} = \langle\langle G, P(\emptyset, a_1), \lceil \rho^{1+\frac{\varepsilon}{2-\varepsilon}} \rceil \rangle\rangle$, where G is a chain long enough to accommodate all paths of $P(\emptyset, a_1)$, is a game in S-PMC(Chain: $L = \Theta(k^{2-\varepsilon})$) with the desired properties. In what follows we briefly sketch the proof of this claim.

In order to prove that the coloring described above is indeed a Nash Eequilibrium for all paths in $P(\emptyset, a_1)$, we first note that for every $P(A, a_i)$ its main and secondary paths lying to the left (resp. right) of the central edge are blocked from switching to any color $a_j \notin A$ (with $a_j \neq a_i$) by the left (resp. right) copy of $P\left(A \cup \{a_i\}, a_j\right)$, where color a_j appears with multiplicity $\rho - |A| - 1$. Furthermore, it is easy to show (using a straightforward induction on the size of A) that they are also blocked from switching to any color $a_j \in A$ by the main and secondary paths of some path set $P(A', a_j)$, with

$P(A', a_j) \succ^* P(A, a_i)$. Since every path of $P(\emptyset, a_1)$ is a main or secondary path for some path set $P(A, a)$, it follows that all paths are in Nash equilibrium. Finally, for the coloring described, the maximum multiplicity appears for color a_1 on the central edge of path set $P(\emptyset, a_1)$ and is indeed equal to ρ.

We then notice that, since the number of colors $k = \left\lceil \rho^{1+\frac{\varepsilon}{2-\varepsilon}} \right\rceil$ exceeds that of the maximum multiplicity $\mu_{\max} = \rho$, the recursive construction of $P(\emptyset, a_1)$ will eventually reach a trivial base case, which incurs an additional load of $\Theta(1)$. It is now easy to show that the maximum load is indeed $L = \Theta(\rho^2)$. This completes the construction.

Now, the class S-PMC(Chain) is a subclass of S-PMC(Rooted-Tree), therefore by Theorem 4.11 the optimal strategy profile for the game defined above has social cost $\mu_{\text{OPT}} = \left\lceil \frac{L}{k} \right\rceil$, hence $\mu_{\text{OPT}} < \frac{L}{k} + 1$. Additionally, the cost of the worst-case Nash equilibrium must be $\hat{\mu} \geq \mu_{\max}$. The price of anarchy for this game is therefore:

$$\text{PoA}(\mathcal{B}) = \frac{\hat{\mu}}{\mu_{\text{OPT}}} > \frac{\mu_{\max}}{\frac{L}{k} + 1} = \frac{k \cdot \mu_{\max}}{L + k} \ . \tag{4.55}$$

It is easy to see that, if $\mu_{\max}$, L, and k satisfy the above properties, then the last expression grows like $\Theta\left(\rho^{\frac{\varepsilon}{2-\varepsilon}}\right)$ as ρ goes to infinity, yielding the promised asymptotic lower bound of $\Omega(k^{\frac{\varepsilon}{2}})$. □

Notice that there is no sense in considering games where the number of wavelengths is larger than the number of paths, i.e. $k \geq |\mathcal{P}|$, because the price of anarchy would then be trivially equal to 1. It is easy to verify that the construction of Theorem 4.26 allows us to create instances with arbitrarily large ratio $\frac{k^2}{L}$, with the number of paths increasing accordingly in any case thus ensuring that $k < |\mathcal{P}|$.

4.8 Conclusions

In this chapter we conducted a thorough study of the price of anarchy in the Selfish Path MultiColoring model for non-cooperative wavelength assignment in multifiber optical networks. The results we obtained in Section 4.6 show that the price of anarchy can grow unbounded even in simple optical networks of maximum degree 3. It seems that the player-charging mechanism of this model is not strong enough to force players into low-fiber-cost Nash equilibria.

On the other hand, the situation regarding networks of degree 2 seems to be rather gratifying. Given the ever-increasing network traffic and the fact that the number of available wavelengths in commercially deployed optical fibers is not expected to grow in the near future, it makes perfect sense to stipulate $k = O\left(\sqrt{L}\right)$ as a valid assumption for practical purposes. Therefore, it is safe to conclude that the price of anarchy of our model is bounded for rings and chains.

We have also shown that when the number of available wavelengths increases beyond $\Theta\left(\sqrt{L}\right)$, it is possible to construct SELFISH PATH MULTI-COLORING games with unbounded price of anarchy. This may be compared to the Braess paradox [17, 18], where the inclusion of a high-speed link in a transportation network results in the deterioration of the solution reached by the players.

Finally, stars, another important network topology, possess low-fiber-cost worst-case Nash equilibria and we can even compute $\frac{1}{2}$-approximate Nash equilibria for games in stars, in view of Theorem 4.12.

Chapter 5

Colored Resource Allocation Games

5.1 Introduction

We now present a general model for non-cooperative resource allocation. In this model, the players have access to a set of resources, each one of which comes in several different versions—one for each color. As is the case in congestion games [67], each player can choose among various subsets of the set of resources. However, the resources picked by a single player must all be of the same color.

This class of games, under various social and player costs, can model non-cooperative versions of routing and wavelength assignment problems in multifiber all-optical networks. These games can be viewed as an extension of congestion games where each player has his strategies in multiple copies corresponding to colors. When restricted to (optical) network games, facilities correspond to physical links of the network and colors correspond to wavelengths. The number of players using an edge in the same color represents a lower bound on the number of fibers needed to implement the corresponding physical link. The wavelength continuity constraint is reflected in the model by the restriction we have placed on player strategy spaces, i.e., that each player must pick resources of the same color. Having this motivation in mind, we consider both the *max* player cost, where the cost of a player is the maximum congestion over the resources she uses, and the *sum* player cost, where the cost of a player is the sum of the congestion of the resources she uses. For our purposes it suffices to restrict our study to identity latency functions.

We estimate the price of anarchy of colored resource allocation games under three different social cost functions. Two of them are standard in the literature (see e.g. [24]): the first (*max* social cost) is equal to the maximum

Table 5.1: The price of anarchy of Colored Congestion Games (*sum* player cost). Results for classical congestion games are shown in the right column.

Social cost	Colored Congestion Games	Congestion Games
$\mathrm{sc}_{\mathrm{max}}$	$\Theta\left(\sqrt{\frac{N}{k}}\right)$	$\Theta\left(\sqrt{N}\right)$ [24]
$\mathrm{sc}_{\mathrm{sum}}$	$\frac{5}{2}$	$\frac{5}{2}$ [24]
$\mathrm{sc}_{\mathrm{fib}}$	$\Theta\left(\sqrt{k \cdot \lvert F \rvert}\right)$	—

Table 5.2: The price of anarchy of Colored Bottleneck Games (*max* player cost). Results for classical bottleneck games are shown in the right column.

Social cost	Colored Bottleneck Games	Bottleneck Games
$\mathrm{sc}_{\mathrm{max}}$	$\Theta\left(\frac{N}{k}\right)$	$\Theta(N)$ [19]
$\mathrm{sc}_{\mathrm{sum}}$	$\Theta\left(\frac{N}{k}\right)$	$\Theta(N)$ [19]
$\mathrm{sc}_{\mathrm{fib}}$	$\Theta\left(\frac{\lvert E_{\vec{A}} \rvert}{\lvert E_{\vec{A}^*} \rvert} \frac{N}{k}\right)$	—

player cost and the second (*sum* social cost) is equal to the sum of player costs or, equivalently, the average player cost. The third one, which we will call *fiber* social cost, is especially meaningful in the setting of multifiber all-optical networks: it is equal to the sum over all facilities of the maximum color congestion on each facility. Note that in the optical network setting this function represents the total fiber cost needed to accommodate the communication requests, hence it captures the objective of a well-studied optimization problem ([62, 61, 3, 4]). Let us also note that the *max* social cost function under the *max* player cost captures the objective of another well known problem, namely minimizing the maximum fiber multiplicity over all edges of the network [3, 49, 53], which we studied in depth in Chapter 4. However, in Chapter 4 we focused on a *fixed routing* model where the path on which each communiciation request is to be routed has been decided in advance, and the only choice given to players was the choice of wavelength. The model we study in this chapter allows players to make routing decisions as well.

We derive tight bounds on the price of anarchy for Colored Resource Allocation Games [9]. These bounds are summarized in Tables 5.1 and 5.2. It can be shown that the bounds for Colored Congestion Games remain tight even for network games.

Observe that known bounds for classical congestion and bottleneck games can be obtained from our results by simply setting $k = 1$. On the other hand one might notice that our games can be casted as classical congestion or bottleneck games with $W|F|$ facilities. However, we are able to derive better upper bounds in some cases by exploiting the special structure of the players' strategies.

5.2 Preliminaries

Definition 5.1 (Colored resource allocation games). *A colored resource allocation game is defined as a tuple $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$, where:*

1. *F is a set of facilities f_i, $1 \le i \le |F|$.*
2. *$\mathcal{P}$ is the set of players. Let N denote the size of set $\mathcal{P}$.*
3. *k is the number of colors. As usual, we will use W for the set of colors.*
4. *$\mathcal{E}_i \subseteq 2^F$ is the set of possible facility combinations for player i. The set $S_i = \mathcal{E}_i \times W$ is the set of available strategies for player i. We will denote the facility combination chosen by player i by $E_i \in \mathcal{E}_i$, and the color choice of player i by $a_i \in W$. We will then say that player i is playing strategy $A_i = (E_i, a_i) \in S_i$.*

A strategy profile for the game will be denoted by a vector $\vec{A} = (A_1, \ldots, A_{|\mathcal{P}|})$ of player strategies. We will use the notation $n_{f,c}\left(\vec{A}\right)$ for the number of players that use facility $f \in F$ with color $c \in W$ in the strategy profile $\vec{A}$.

Depending on the player cost function we define two subclasses of colored resource allocation games:

Definition 5.2 (Colored Congestion Games). *A Colored Congestion Game (CCG) is a colored resource allocation game with sum player costs, defined as follows for each player $i \in \mathcal{P}$:*

$$C_i\left(\vec{A}\right) = \sum_{e\in E_i} n_{e,a_i}\left(\vec{A}\right) \quad . \tag{5.1}$$

Definition 5.3 (Colored Bottleneck Games). *A Colored Bottleneck Game (CBG) is a colored resource allocation game with max player costs, defined as follows for each player $i \in \mathcal{P}$:*

$$C_i\left(\vec{A}\right) = \max_{e\in E_i} n_{e,a_i}\left(\vec{A}\right) \quad . \tag{5.2}$$

In the same manner as we did in Chapter 4, we will say that a strategy profile is a *pure Nash equilibrium* (PNE), or simply *Nash equilibrium* (NE), if no player can reduce her cost by changing strategy unilaterally. From the definition of Nash Equilibrium we can derive the following two facts that hold in Colored Congestion and Bottleneck Games.

Fact 5.4. *If $\vec{A}$ is a Nash equilibrium of a CCG game, then for any player $i \in \mathcal{P}$ we have:*

$$C_i\left(\vec{A}\right) \le \sum_{e \in E'_i} \left(n_{e,a'_i}\left(\vec{A}\right) + 1\right) \quad , \tag{5.3}$$

for any $E'_i \in \mathcal{E}_i$ and for any $a'_i \in W$.

Fact 5.5. *If $\vec{A}$ is a Nash equilibrium of a CBG game, then for any player $i \in \mathcal{P}$ we have:*

$$C_i\left(\vec{A}\right) \le \max_{e \in E'_i} \left(n_{e,a'_i}\left(\vec{A}\right) + 1\right) \quad , \tag{5.4}$$

for any $E'_i \in \mathcal{E}_i$ and for any $a'_i \in W$. Equivalently, for any $E'_i \in \mathcal{E}_i$ and $a'_i \in W$, there is some $e \in E'_i$ such that

$$C_i\left(\vec{A}\right) \le n_{e,a'_i}\left(\vec{A}\right) + 1 \quad . \tag{5.5}$$

We note immediately the similarity of Equation 5.5 to Equation 4.9 which defines the blocking edges in Definition 4.5.

For each one of the two subclasses of colored resource allocation games that we just defined, we will consider three different social cost functions. The *max* player cost $\mathrm{sc}_{\max}$ is given by

$$\mathrm{sc}_{\max}\left(\vec{A}\right) = \max_{i \in \mathcal{P}} C_i\left(\vec{A}\right) \quad . \tag{5.6}$$

The *sum* player cost $\mathrm{sc}_{\mathrm{sum}}$ is given by

$$\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}\right) = \sum_{i \in \mathcal{P}} C_i\left(\vec{A}\right) \quad . \tag{5.7}$$

Finally, the *fiber* social cost $\mathrm{sc}_{\mathrm{fib}}$ is given by

$$\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}\right) = \sum_{f \in F} \max_{a \in W} n_{f,a}\left(\vec{A}\right) \quad . \tag{5.8}$$

Let $\vec{A}^*$ be a minimum-social-cost strategy profile for some colored resource allocation game $\mathcal{G}$ under some social cost sc. Analogously to the definition in Section 4.3.1, the *price of anarchy* (PoA) of $\mathcal{G}$ is:

$$\mathrm{PoA}\left(\mathcal{G}\right) = \frac{\max_{\vec{A} \text{ is NE}} \mathrm{sc}\left(\vec{A}\right)}{\mathrm{sc}\left(\vec{A}^*\right)} \quad . \tag{5.9}$$

5.3 Colored Congestion Games

5.3.1 The Price of Anarchy for *max* Social Cost

Theorem 5.6. *Under the social cost* $\mathrm{sc}_{\max}$*, the price of anarchy of any Colored Congestion Game* $\langle\!\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\!\rangle$ *is* $O\left(\sqrt{\frac{N}{k}}\right)$.

Proof. Let $\vec{A}$ be a Nash equilibrium and let $\vec{A}^*$ be an optimal strategy profile. Without loss of generality we consider the first player to have the maximum cost, $\mathrm{sc}_{\max}\left(\vec{A}\right) = C_1\left(\vec{A}\right)$. Thus, we need to bound $C_1\left(\vec{A}\right)$ with respect to the optimum social cost $\mathrm{sc}_{\max}\left(\vec{A}^*\right) = \max_{j\in\mathcal{P}} C_j\left(\vec{A}^*\right)$.

Let $\vec{A}^*_1 = (E^*_1, a^*_1)$ be the strategy of player 1 in $\vec{A}^*$. Since $\vec{A}$ is a Nash equilibrium, no player benefits by changing either her color or her choice of facilities. Therefore, for any $a \in W$:

$$C_1\left(\vec{A}\right) \le \sum_{e\in E^*_1}\left(n_{e,a}\left(\vec{A}\right) + 1\right) \le \sum_{e\in E^*_1} n_{e,a}\left(\vec{A}\right) + C_1\left(\vec{A}^*\right) \ . \tag{5.10}$$

Let $I \subseteq \mathcal{P}$ be the set of players that use some facility $e \in E^*_1$ in strategy profile $\vec{A}$. The sum of their costs in $\vec{A}$ is:

$$\begin{aligned}
\sum_{i\in I} C_i\left(\vec{A}\right) &\ge \sum_{e\in E^*_1}\sum_{a\in W} n^2_{e,a}\left(\vec{A}\right) && (5.11)\\
&\ge \frac{\left(\sum_{e\in E^*_1}\sum_{a\in W} n_{e,a}\left(\vec{A}\right)\right)^2}{\left|E^*_1\right|\cdot k} && (5.12)\\
&\ge \frac{\left(k\cdot \min_{a\in W}\sum_{e\in E^*_1} n_{e,a}\left(\vec{A}\right)\right)^2}{\left|E^*_1\right|\cdot k} && (5.13)\\
&\ge \frac{k\cdot\left(\min_{a\in W}\sum_{e\in E^*_1} n_{e,a}\left(\vec{A}\right)\right)^2}{\left|E^*_1\right|} \ . && (5.14)
\end{aligned}$$

The first inequality holds since a player in I might also use facilities not in E^*_1, and the second inequality holds from the Cauchy-Schwarz inequality. Let $a_{\min} = \arg\min_{a\in W}\sum_{e\in E^*_i} n_{e,a}\left(\vec{A}\right)$. Thus we have:

$$\left(\sum_{e\in E^*_1} n_{e,a_{\min}}\left(\vec{A}\right)\right)^2 \le \frac{\left|E^*_1\right|}{k}\cdot\sum_{i\in I} C_i\left(\vec{A}\right) \ . \tag{5.15}$$

From [24] we have:

$$\sum_{i\in\mathcal{P}} C_i\left(\vec{A}\right) \le \frac{5}{2}\sum_{i\in\mathcal{P}} C_i\left(\vec{A}^*\right) \ . \tag{5.16}$$

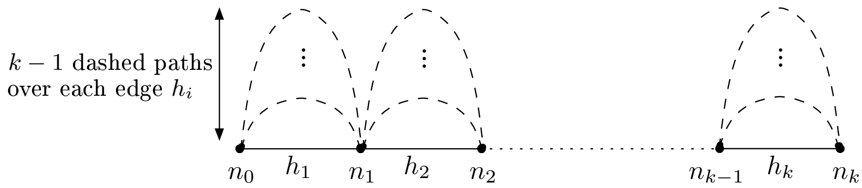

Figure 5.1: A worst-case instance that proves the tightness of the upper bound, depicted as network game. A dashed line represents a path of length ℓ connecting its two endpoints.

Combining the above two inequalities we get:

$$\left(\sum_{e\in E_1^*} n_{e,a_{\min}}\left(\vec{A}\right)\right)^2 \leq \frac{\left|E_1^*\right|}{k}\sum_{i\in I} C_i\left(\vec{A}\right) \leq \frac{\left|E_1^*\right|}{k}\sum_{i\in\mathcal{P}} C_i\left(\vec{A}\right) \leq \frac{5}{2}\frac{\left|E_1^*\right|}{k}\sum_{i\in\mathcal{P}} C_i\left(\vec{A}^*\right) \ . \tag{5.17}$$

Combining this with Equation 5.10 for $a = a_{\min}$, we get:

$$C_1\left(\vec{A}\right) \leq C_1\left(\vec{A}^*\right) + \sqrt{\frac{5}{2}\frac{\left|E_1^*\right|}{k}\sum_{i\in\mathcal{P}} C_i\left(\vec{A}^*\right)} \ . \tag{5.18}$$

Since $\left|E_1^*\right| \leq C_1\left(\vec{A}^*\right)$ and $C_i\left(\vec{A}^*\right) \leq \mathrm{sc}_{\max}\left(\vec{A}^*\right)$, we finally get

$$C_1\left(\vec{A}\right) \leq \left(1 + \sqrt{\frac{5}{2}\frac{N}{k}}\right)\mathrm{sc}_{\max}\left(\vec{A}^*\right) \ . \tag{5.19}$$

□

Theorem 5.7. *There exists an infinite family of Colored Congestion Games* $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$ *with social cost* $\mathrm{sc}_{\max}$, *that have price of anarchy* $\Omega\left(\sqrt{\frac{N}{k}}\right)$.

Proof. We will describe the lower bound game as a network game. The underlying network is illustrated in Figure 5.1. The game itself is a small variation of the construction presented in [24].

In that network k major players want to send traffic from n_0 to n_ℓ. For every i, $0 \leq i \leq \ell - 1$, there are $(\ell - 1)k$ minor players that want to send traffic from node n_i to node n_{i+1}. In the worst-case equilibrium $\vec{A}$ all players choose the short central edge, leading to social cost $\mathrm{sc}_{\max}\left(\vec{A}\right) = \ell^2$. On the other hand, in the optimum strategy profile $\vec{A}^*$, the minor players are equally divided on the dashed-line paths and the major players choose the central edge. This leads to $\mathrm{sc}_{\max}\left(\vec{A}^*\right) = \ell$, and the price of anarchy is therefore:

$$\mathrm{PoA}\left(\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle\right) = \ell = \Theta\left(\sqrt{\frac{N}{k}}\right) . \tag{5.20}$$

□

5.3.2 The Price of Anarchy for *sum* Social Cost

The price of anarchy for social cost $\mathrm{sc}_{\mathrm{sum}}$ is upper-bounded by $\frac{5}{2}$, as proved in [24]. For the lower bound, we use a slight modification of the construction described in [24]. We have Nk players and $2N$ facilities. The facilities are separated into two groups: $\{h_1, \ldots, h_N\}$ and $\{g_1, \ldots, g_N\}$. The players are divided into N groups of k players. Each group i has possible facility combinations $\{h_i, g_i\}$ and $\{g_{i+1}, h_{i-1}, h_{i+1}\}$. The optimal strategy profile $\vec{A}^*$ is for all players in the i-th group to select their first strategy and be equally divided in the k colors, leading to $\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}^*\right) = 2Nk$. In the worst-case Nash equilibrium $\vec{A}$ players choose their second strategy and are equally divided in the k colors, leading to $\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}\right) = 5Nk$. Thus, the price of anarchy of this game is $\frac{5}{2}$ and the upper bound remains tight in our model too.

5.3.3 The Price of Anarchy for *fiber* Social Cost

Theorem 5.8. *Under the social cost* $\mathrm{sc}_{\mathrm{fib}}$*, the price of anarchy of any Colored Congestion Game* $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$ *is* $O\left(\sqrt{k\cdot|F|}\right)$.

Proof. Let $\vec{S}$ be a strategy profile of the game and $W = \{w_1, \ldots, w_k\}$ be the set of colors. For a facility $e \in F$, we denote by $\overline{n_e\left(\vec{S}\right)}$ the vector

$$\overline{n_e\left(\vec{S}\right)} = \left(n_{e,w_1}\left(\vec{S}\right), \ldots, n_{e,w_k}\left(\vec{S}\right)\right) . \tag{5.21}$$

In terms of the above vector we can write:

$$\mathrm{sc}_{\mathrm{fib}}\left(\vec{S}\right) = \sum_{e\in F} \max_{a\in W} n_{e,a}\left(\vec{S}\right) = \sum_{e\in F} \left\|\overline{n_e\left(\vec{S}\right)}\right\|_\infty . \tag{5.22}$$

From norm inequalities we have that

$$\frac{\left\|\overline{n_e\left(\vec{S}\right)}\right\|_2}{\sqrt{k}} \le \left\|\overline{n_e\left(\vec{S}\right)}\right\|_\infty \le \left\|\overline{n_e\left(\vec{S}\right)}\right\|_2 , \tag{5.23}$$

therefore:

$$\mathrm{sc}_{\mathrm{fib}}\left(\vec{S}\right) = \sum_{e\in F} \left\| \overline{n_e\left(\vec{S}\right)} \right\|_\infty \leq \sum_{e\in F} \sqrt{\sum_{a\in W} n_{e,a}^2\left(\vec{S}\right)} \leq \sqrt{|F|}\sqrt{\sum_{e\in F}\sum_{a\in W} n_{e,a}^2\left(\vec{S}\right)} \ , \tag{5.24}$$

where the last inequality is a manifestation of the norm inequality $\|\vec{x}\|_1 \leq \sqrt{n}\,\|\vec{x}\|_2$, where $\vec{x}$ is a vector of dimension n. Now, from the first inequality of Equation 5.23 we have:

$$\mathrm{sc}_{\mathrm{fib}}\left(\vec{S}\right) \geq \frac{1}{\sqrt{k}} \sum_{e\in F} \sqrt{\sum_{a\in W} n_{e,a}^2\left(\vec{S}\right)} \geq \frac{1}{\sqrt{k}} \sqrt{\sum_{e\in F}\sum_{a\in W} n_{e,a}^2\left(\vec{S}\right)} \ . \tag{5.25}$$

Combining Equations 5.25 and 5.24 yields:

$$\frac{1}{\sqrt{k}} \sqrt{\mathrm{sc}_{\mathrm{sum}}\left(\vec{S}\right)} \leq \mathrm{sc}_{\mathrm{fib}}\left(\vec{S}\right) \leq \sqrt{|F|}\cdot\sqrt{\mathrm{sc}_{\mathrm{sum}}\left(\vec{S}\right)} \ . \tag{5.26}$$

From [24] we know that the price of anarchy under the *sum* social cost is $\frac{5}{2}$. Let $\vec{A}$ be a worst-case Nash equilibrium under the *fiber* social cost and let $\vec{A}^*$ be an optimal strategy profile under the same social cost. From Equation 5.26 we know that $\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}\right) \leq \sqrt{|F|}\cdot\sqrt{\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}\right)}$ and $\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}^*\right) \geq \frac{1}{\sqrt{k}}\sqrt{\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}^*\right)}$. Thus:

$$\mathrm{PoA}\left(\langle\!\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\!\rangle\right) = \frac{\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}\right)}{\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}^*\right)} \leq \sqrt{k\cdot|F|}\sqrt{\frac{\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}\right)}{\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}^*\right)}} \leq \sqrt{k\cdot|F|}\sqrt{\frac{5}{2}} \ . \tag{5.27}$$

□

Theorem 5.9. *There exists an infinite family of Colored Congestion Games* $\langle\!\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\!\rangle$ *with social cost* $\mathrm{sc}_{\mathrm{fib}}$, *that have price of anarchy* $\sqrt{k\cdot|F|}$.

Proof. Consider a Colored Congestion Game with N players, $|F| = N$ facilities and $k = N$ colors. The possible facility combinations for each player consist of all singleton facility sets, i.e., $\mathcal{E}_i = \{\{f_1\}, \{f_2\}, \ldots, \{f_N\}\}$.

The above game has a worst-case Nash equilibrium with social cost N when all players choose a different facility in arbitrary colors. On the other hand, in the optimum strategy profile the players use as few facilities as possible, filling up all colors of these facilities. This requires $\frac{N}{k}$ facilities, each of which contributes 1 to the social cost. Therefore, the optimum social cost is $\frac{N}{k}$, yielding a price of anarchy of $k = \sqrt{k\cdot|F|}$. □

5.4 Colored Bottleneck Games

5.4.1 The Price of Anarchy for *max* Social Cost

Theorem 5.10. *Under the social cost* $\mathrm{sc}_{\max}$*, the price of anarchy of any Colored Bottleneck Game* $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$ *is at most* $\left\lceil \frac{N}{k} \right\rceil$.

Proof. Assume that the social cost of a Nash equilibrium is strictly larger than $\left\lceil \frac{N}{k} \right\rceil$. This implies that there is some player whose cost is strictly larger than $\left\lceil \frac{N}{k} \right\rceil$, so there is some facility-color pair used by strictly more than $\left\lceil \frac{N}{k} \right\rceil$ players. Since we are in Nash equilibrium, Fact 5.5 implies that for every other color there must be some facility in this player's chosen facility combination used by strictly more than $\left\lceil \frac{N}{k} \right\rceil - 1$ players. Since players playing different colors are necessarily distinct, the above implies the existence of at least $(k-1) \cdot \left\lceil \frac{N}{k} \right\rceil + \left\lceil \frac{N}{k} \right\rceil + 1 \geq N + 1$ players, which is a contradiction.

Therefore, the social cost of any Nash equilibrium is at most $\left\lceil \frac{N}{k} \right\rceil$. Since the minimum-cost strategy profile has cost at least 1, the claim is proved. □

Theorem 5.11. *There exists an infinite family of Colored Bottleneck Games* $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$ *with social cost* $\mathrm{sc}_{\max}$*, that have price of anarchy* $\frac{N}{k}$.

Proof. Consider the following class of CBG games. There are N players, N facilities, and k colors, where N is an integer multiple of k. Each player i has two possible facility combinations: $\mathcal{E}_i = \{\{f_i\}, \{f_1, \ldots, f_N\}\}$. In a worst-case Nash equilibrium, all players choose the second combination and they are equally divided in the k colors. This leads to a player cost of $\frac{N}{k}$ for each player and thus to a social cost of $\frac{N}{k}$. In the optimal strategy profile, all players would choose their first strategy leading to player cost and social cost equal to 1. Thus, the price of anarchy for this game is $\frac{N}{k}$. □

5.4.2 The Price of Anarchy for *sum* Social Cost

By Theorem 5.10, we know that in any Nash equilibrium $\vec{A}$, the cost of any player is $C_i\left(\vec{A}\right) \leq \left\lceil \frac{N}{k} \right\rceil$. Moreover, it is not hard to see that in the minimum-cost strategy profile $\vec{A}^*$, $\mathrm{sc}_{\mathrm{sum}}\left(\vec{A}^*\right) \geq N$. Therefore, the price of anarchy is upper-bounded by $\frac{N \cdot \left\lceil \frac{N}{k} \right\rceil}{\mathrm{sc}_{\mathrm{sum}}(\vec{A}^*)} \leq \left\lceil \frac{N}{k} \right\rceil$.

The game we constructed in the proof of Theorem 5.11 can also be used here to prove that the bound we just proved is tight.

5.4.3 The Price of Anarchy for *fiber* Social Cost

Definition 5.12. *Let $\vec{S}$ be a strategy profile of a game $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$. We define $E_{\vec{S}}$ to be the set of facilities used by at least one player in the strategy profile $\vec{S}$, i.e.*

$$E_{\vec{S}} = \bigcup_{i\in\mathcal{P}} E_i \ . \tag{5.28}$$

Theorem 5.13. *Under the social cost $\mathrm{sc}_{\mathrm{fib}}$, the price of anarchy of any Colored Bottleneck Game $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$ is at most $\frac{|E_{\vec{A}}|}{|E_{\vec{A}^*}|} \cdot \left\lceil \frac{N}{k} \right\rceil$, where $\vec{A}$ is a worst-case Nash equilibrium and $\vec{A}^*$ is a minimum-cost strategy profile.*

Proof. Let $a_{\max}(e)$ denote the color with the maximum multiplicity at facility e, in the worst-case Nash equilibrium $\vec{A}$. Let i be a player that uses the facility copy $(e, a_{\max}(e))$. Since $C_i\left(\vec{A}\right) = \max_{e\in E_i} n_{e,a_i}\left(\vec{A}\right)$, it must hold that $n_{e,a_{\max}(e)}\left(\vec{A}\right) \le C_i\left(\vec{A}\right)$. In fact we can state the following general property: for every $e \in F$, there is some player i such that

$$n_{e,a_{\max}(e)}\left(\vec{A}\right) \le C_i\left(\vec{A}\right) \ . \tag{5.29}$$

We already know from Theorem 5.10 that for any player i, $C_i\left(\vec{A}\right) \le \left\lceil \frac{N}{k} \right\rceil$, since $\vec{A}$ is a Nash equilibrium. Moreover, it is clear that $\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}^*\right) \ge |E_{\vec{A}^*}|$. From the above we can conclude:

$$\frac{\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}\right)}{\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}^*\right)} \le \frac{|E_{\vec{A}}|}{|E_{\vec{A}^*}|} \cdot \left\lceil \frac{N}{k} \right\rceil \ . \tag{5.30}$$

□

Theorem 5.14. *There exists an infinite family of Colored Bottleneck Games $\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle$ with social cost $\mathrm{sc}_{\mathrm{fib}}$, that have price of anarchy $\frac{|E_{\vec{A}}|}{|E_{\vec{A}^*}|} \cdot \left\lceil \frac{N}{k} \right\rceil$, where $\vec{A}$ is a worst-case Nash equilibrium and $\vec{A}^*$ is a minimum-cost strategy profile.*

Proof. We use a slight modification of the game used in Theorem 5.11. The set of possible facility combinations of each player i is $\mathcal{E}_i = \{\{f_i\}, \{f_1, \ldots, f_M\}\}$, for some $M \ge N$. In the worst-case Nash equilibrium all players will play the second combination leading to $\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}\right) = M \cdot \frac{N}{k}$ and $|E_{\vec{A}}| = M$. On the other hand, in the minimum-cost strategy profile, all players will play the first combination leading to $\mathrm{sc}_{\mathrm{fib}}\left(\vec{A}^*\right) = N$ and $|E_{\vec{A}^*}| = N$. Therefore, the price of anarchy for this game is $\mathrm{PoA}\left(\langle\langle F, \mathcal{P}, k, \{\mathcal{E}_i\}_{i\in\mathcal{P}}\rangle\rangle\right) = \frac{M}{N} \cdot \frac{N}{k} = \frac{|E_{\vec{A}}|}{|E_{\vec{A}^*}|} \cdot \frac{N}{k}$. □

5.5 Conclusions

We have introduced colored resource allocation games, a class of games which generalize both congestion and bottleneck games. The main feature of these games is that players have their strategies in multiple copies (colors). Therefore, these games can serve as a framework to describe routing and wavelength assignment games in multifiber all-optical networks. Although one might cast such games as classical congestion or bottleneck games, it turns out that the proliferation of resources together with the structure imposed on the players' strategies allows us in some cases to prove better bounds on the price of anarchy.

In our exposition, we have not considered the question of *convergence* to Nash equilibria. Let us briefly note here that Colored Congestion Games admit of exact potentials, since, as we said, they can be cast as regular Congestion Games. Therefore, their convergence to a Nash equilibrium in finitely many steps is guaranteed. In the case of Colored Bottleneck Games, one can follow lexicographic-order arguments similar to the ones used in the proof of Theorem 4.8, in order to show that these games also admit a potential, albeit not an exact one. Therefore, convergence to a Nash equilibrium is guaranteed in this case as well.

Regarding the bounds that we managed to show on the price of anarchy, one may say that these are mostly negative results. With the exception of games with *sum* social and player costs where the price of anarchy is constant, in all other cases the price of anarchy is unbounded. Especially in the case of Colored Congestion Games, all of our lower bound constructions can be recast as network games. Therefore, we know there exist actual networks where the price of anarchy can be unbounded. On the other hand, the constructions we give for Colored Bottleneck Games cannot be seen as network games. This seems to give some hope that a more detailed study of Colored Bottleneck Games defined on networks might yield better bounds on the price of anarchy.

In both cases, it would be interesting to examine specific network topologies and see which of them allow for better system behavior. Finally, another direction would be to consider more general latency functions. This would make sense both in the case where fiber pricing is not linear in the number of fibers, and also in the case where the network operator seeks to determine an appropriate pricing policy so as to reduce the price of anarchy.

Chapter 6

Path Coloring Applied to a Transportation Problem

In this chapter we deal with an interesting problem in transportation networks, that of scheduling a given set of routes on a transportation network so that the minimum headway is maximized. The minimum headway is a measure of the flexibility of the schedule with respect to perturbations of the departure or arrival times of a small number of routes. The exposition in this chapter aims to bring forward a fundamental connection between the headway maximization problem and a path coloring problem that has been thoroughly studied in the literature in view of its direct application on wavelength assignment in WDM networks.

The purpose of this chapter is twofold. First, we study the complexity of headway optimization and provide exact and approximation algorithms for certain fundamental network topologies. Second, our arguments are permeated with the aforementioned underlying connection to path coloring, which serves to highlight the applicability of path coloring models on a wide and diverse range of network/scheduling problems, apart from the routing and wavelength assignment problems in WDM networks which have been our main focus throughout this thesis.

6.1 Introduction

In railway networks where trains use the same railway segment quite often (e.g. metro) it would be desirable to schedule trains so as to guarantee an ample time distance between successive trains that pass from the same point of the network (in the same direction); this time distance is usually called *headway*. Such a scheduling would result in a more delay-tolerant system. This is a particularly essential requirement in cases where there are several intersecting routes that have to be carried out periodically and the time limits are such that some route must start before the termination of another route with which it shares a part of the network.

Here, we formulate this situation by introducing the PERIODIC METRO SCHEDULING (PMS) problem: given a rail network with one line per direction, a set of routes (described as paths over the network graph), and a time period, we seek to arrange the departure times of routes so that the minimum headway is maximized. Although our motivation comes from railway optimization, PMS may also describe other transportation media timetabling problems.

We show [7] the NP-hardness of PMS by reduction from PATH COLORING (PC), which is the problem of coloring paths in a graph with the minimum number of colors so that intersecting paths receive different colors. We further investigate the relation between the two problems and present exact algorithms for chain and spider networks that rely on a reverse reduction from PMS to PC. Moreover, we show that this technique also applies to rings for which the time needed to traverse the ring is a multiple of the given period. This results in a $\left(\frac{1}{\rho}\frac{L}{L+1}\right)$-approximation algorithm for such instances, where ρ is the approximation ratio we can achieve for PC and L is the maximum number of routes passing through any edge of the network. For ring instances that do not satisfy this condition we present a specifically designed algorithm that achieves an approximation guarantee of $\frac{1}{5}$. Finally, we show how to apply the path coloring technique to tree networks where the time distances between stations are integer multiples of the half of the period, resulting in a $\left(\frac{1}{\rho}\frac{L}{L+1}\right)$-approximation algorithm for this topology as well. Our algorithms employ known algorithms for PC [22, 55, 34, 39] as subroutines.

6.2 Related Work

To the best of our knowledge, PERIODIC METRO SCHEDULING has not been studied before in the form of an optimization problem. The decision version of PMS, namely the problem of guaranteeing a minimum headway not smaller than a given threshold, can be described in terms of a generic problem known as PERIODIC EVENT SCHEDULING PROBLEM (PESP) [72]. PESP has been studied by several researchers, see e.g. [71, 51, 47, 50] and references therein. However, we are not aware of any concrete results for PESP that could apply to PMS, as PESP is usually studied in conjuction with several other constraints that render the problem quite hard and the proposed methods for solving it are mainly algorithms with no guaranteed efficiency based on "branch-and-bound", "branch-and-cut", and "branch-and-price" methods. A similar problem as PMS has been defined in [43], and it has been proven to be NP-complete. However, the setting is again broader and the completeness results apply to general graphs.

There is a huge bibliography on railway optimization topics; the interested reader is referred to [20] for a nice collection of concepts and earlier results on railway optimization. More recent work on periodic train scheduling includes a rolling stock minimization problem where routes are given and it is sought to determine departure times either arbitrarily (as in our case) or within an allowed time window [30]. However, the objective there is quite different, namely to serve all routes with a minimum number of trains while it is allowed for routes to simultaneously depart from the same station even if they follow the same direction (it is assumed that multiples lines are available). The rolling stock minimization problem with fixed departure times has been extensively studied: the simplest version, also known as MINIMUM FLEET SIZE [14], or ROLLING STOCK ROSTERING [31], can be solved exactly in polynomial time; Dantzig and Fulkerson [27] give the first known algorithm and Erlebach et al. [31] present one of improved complexity. In [31], certain variations are also studied and shown APX-hard: allowing empty rides and requiring that the trains pass through a maintenance station.

Another problem that has recently drawn attention is that of delay management, that is, the question of how to reduce or increase delays of trains in order to better serve railway customers [70, 40, 41].

Very recently, Dahl [26] has studied a model which is quite closely related to PERIODIC METRO SCHEDULING. It can be viewed as a special case of PMS in which the routes only share a single edge and the question is whether the frequency of each route can be adjusted so that there are no collisions.

6.3 Preliminaries

We assume that all trains move at the same speed, therefore the duration of traveling between any two connected stations is the same for all trains. In the sequel we denote the travel time between two stations connected by edge e as $t(e)$. We also assume that edges represent directed railway lines and any two connected stations are linked by a pair of opposite directed edges. For simplicity we consider that the waiting time at stations is negligible.

We are interested in maximizing the minimum headway between any two intersecting routes, that is, routes that share at least one edge. Note that, due to the uniformity of speed, it suffices to measure the headway between intersecting routes only at the starting node (station) of the first edge of each common section. More precisely, let e be a common edge between routes r and r' and t (resp. t') be the time at which r (resp. r') enters edge e. Then, the headway between r and r' at edge e is defined as $\min\{t - t' \bmod T, t' - t \bmod T\}$. When the headway between two routes in an edge is 0, we say that the routes *collide*.

We will denote the source node of a route r by $s(r)$, and its target node by $e(r)$. We define $\tau(i, j)$ to be the time distance between nodes i and j in the input graph, whenever it is uniquely defined.

Let us now formally define the headway maximization problem.

Problem 6.1 (Periodic Metro Scheduling, PMS)**.**
Instance: $\langle G, t, T, \mathcal{R}\rangle$, *where* $G = (V, E)$ *is a directed graph with bidirected edges,* $t : E \to \mathbb{N}$ *is an inter-station time distance function,* $T > 0$ *is an integer time period, and* $\mathcal{R} = \{r_1, \ldots, r_{|\mathcal{R}|}\}$ *is a collection of simple paths defined on* G *(routes).*
Feasible solution: A schedule for $\mathcal{R}$*, that is, a function* $\mathrm{stime} : \mathcal{R} \to [0, T)$ *which assigns a departure time to each route.*
Goal: Maximize the minimum headway between any two intersecting routes.

We define $L(e)$ to be the *congestion* on edge e, that is the number of routes that pass through edge e of the network. Let $L = \max_e L(e)$. It is not hard to see that $\frac{T}{L}$ is an upper bound to the objective value of an optimal solution (OPT), because routes cannot be spaced further apart than $\frac{T}{L}$ on the edge with maximum congestion.

In our study we will pinpoint a close relationship between PMS and Path Coloring (PC), which is defined as follows:

Problem 6.2 (Path Coloring, PC)**.**
Instance: $\langle G, \mathcal{P}\rangle$, *where* G *is a directed graph and* $\mathcal{P} = \{p_1, \ldots, p_{|\mathcal{P}|}\}$ *is a collection of simple paths defined on* G.
Feasible solution: An assignment of colors to all paths of $\mathcal{P}$ *such that intersecting paths are assigned different colors.*
Goal: Minimize the number of colors used.

Path Coloring (note that here we consider the directed version) can be solved optimally in polynomial time in chains, stars, and spiders by a greedy algorithm (folklore, see e.g. [42]) using L colors, but is known to be NP-hard in rings [39] and trees [55]. Note that a *spider* is a tree in which

Algorithm 10 An algorithm for PMS in chain networks

Input: an instance $\langle G, t, T, \mathcal{R} \rangle$ of PMS, where G is a chain

1: Compute a coloring of routes with exactly L colors from $\{0, \ldots, L-1\}$, using the greedy algorithm for PC in chains. Let color(r) denote the color assigned to route r.

2: Set $t = \frac{T}{L}$ and define L time slots as follows: $0, t, 2t, \ldots, (L-1)t$.

3: Assign time slots to routes according to the coloring obtained in step 1, namely: $\mathrm{timeslot}(r) := \mathrm{color}(r) \cdot t$.

4: For each route $r \in \mathcal{R}$ set the starting time as follows:

$$\mathrm{stime}(r) = (\mathrm{timeslot}(r) + \tau(0, s(r))) \bmod T \ .$$

at most one internal node has degree 3 or greater (this is called the *central* node), or equivalently, a graph resulting from a star whose edges have been replaced by chains, also called *legs* of the spider.

We will use the notation $a \equiv_T b$ to denote the fact that $a \bmod T = b \bmod T$.

6.4 Headway Optimization in Chain, Star, and Spider Networks

6.4.1 An Algorithm for Chains

In chains we label the nodes of the graph from 0 to $n-1$ successively. Since all connections are bidirectional we can divide any problem instance into two subproblems, one containing routes moving to the right and one containing routes moving to the left and solve them separately.

Let t_i be the time distance from node i to $i+1$ for $i = 1, \ldots, n-1$. In the case of chain networks, the time distance between two nodes i and j is therefore:

$$\tau(i, j) = \sum_{k=i}^{j-1} t_k \ . \tag{6.1}$$

Our algorithm for PMS in chains makes use of the fact that PC can be solved optimally for chain networks. The description of the algorithm is presented in Algorithm 10.

Theorem 6.3. *Algorithm 10 computes an optimal solution for* PMS *in chains.*

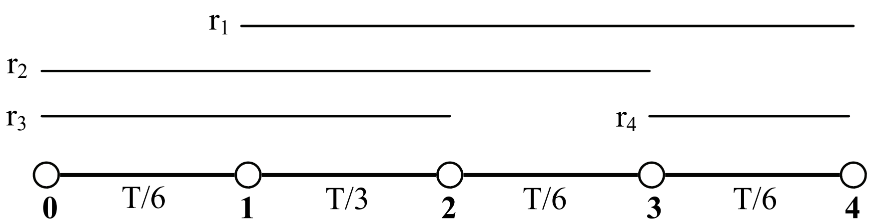

Figure 6.1: An instance of PMS on a chain network.

Proof. Let r and r' be two intersecting routes and without loss of generality assume that $s(r') \leq s(r)$. The first point of their common section is $s(r)$ and their headway at $s(r)$ is:

$$d(r, r', s(r)) = \min\{(\text{stime}(r') + \tau(s(r'), s(r)) - \text{stime}(r)) \bmod T, \\ (\text{stime}(r) - (\text{stime}(r') + \tau(s(r'), s(r)))) \bmod T\} \quad . \tag{6.2}$$

Note that

$$\text{stime}(r') + \tau(s(r'), s(r)) - \text{stime}(r) = \text{timeslot}(r') + \\ \tau(0, s(r')) + \tau(s(r'), s(r)) - (\text{timeslot}(r) + \tau(0, s(r))) = \\ \text{timeslot}(r') - \text{timeslot}(r) \quad , \tag{6.3}$$

therefore

$$d(r, r', s(r)) = \min\{(\text{timeslot}(r') - \text{timeslot}(r)) \bmod T, \\ (\text{timeslot}(r) - \text{timeslot}(r')) \bmod T\} \quad . \tag{6.4}$$

Since the difference between any two time slots is at least $\frac{T}{L}$ and $\frac{T}{L}$ is an upper bound for the value of any feasible solution, the solution returned by the algorithm is optimal. □

Example 6.4. *Consider the instance illustrated in Figure 6.1. The maximum congestion is* $L = 3$ *and as a result the path coloring algorithm will yield a solution with* 3 *colors. The time slots corresponding to these colors are:* 0, $\frac{T}{3}$ *and* $\frac{2T}{3}$. *Assume that routes* r_3 *and* r_4 *are assigned time slot* 0, *route* r_2 *is assigned time slot* $\frac{T}{3}$, *and route* r_1 *is assigned time slot* $\frac{2T}{3}$. *According to Algorithm 10,* $\text{stime}(r_1) = \frac{5T}{6}$, $\text{stime}(r_2) = \frac{T}{3}$, $\text{stime}(r_3) = 0$, *and* $\text{stime}(r_4) = \frac{2T}{3}$. *Observe that on edge* $(1, 2)$ *the three intersecting routes* r_1, r_2, *and* r_3 *have headway at least* $\frac{T}{3}$, *which is optimal. Furthermore, route* r_1 *reaches node* 3 *at time* $\frac{T}{3}$ *("wrapping around" the end of the time period), thus also having headway* $\frac{T}{3}$ *from* r_4.

Algorithm 11 An algorithm for PMS in spider networks

Input: an instance $\langle G, t, T, \mathcal{R} \rangle$ of PMS, where G is a spider

1: Compute a path coloring of routes with exactly L colors from $\{0, \ldots, L-1\}$. Let color(r) denote the color assigned to route r.

2: Set $t = \frac{T}{L}$ and define L time slots as follows: $0, t, 2t, \ldots, (L-1)t$.

3: Assign time slots to routes according to the coloring obtained in step 1, namely timeslot(r) := color(r) $\cdot\, t$.

4: For each route r passing through the central node, set starting time

$$\mathrm{stime}(r) = (\mathrm{timeslot}(r) - \tau(0, s(r))) \bmod T \ .$$

5: For each route r confined in a single leg and directed towards the central node, set starting time

$$\mathrm{stime}(r) = (\mathrm{timeslot}(r) - \tau(0, s(r))) \bmod T \ .$$

6: For each route r confined in a single leg and directed away from the central node, set starting time

$$\mathrm{stime}(r) = (\mathrm{timeslot}(r) + \tau(0, s(r))) \bmod T \ .$$

6.4.2 An Algorithm for Stars and Spiders

Given an instance of PMS on a star or a spider, we will utilize an optimal path coloring of the given instance in order to produce an optimal time schedule. Note that an optimal path coloring can be computed by an exact algorithm for spiders which can be obtained by appropriate combination of a known exact algorithm for stars and the greedy algorithm for chains. We should note that some routes may be confined in one of the spider's legs while others may be directed from one leg to another.

Theorem 6.5. *Algorithm 11 computes an optimal solution for* PMS *in spiders.*

Proof. We will first prove the claim for the case where the spider is a star. Let r and r' be two intersecting routes. Therefore they receive different colors, hence also different time slots. There are two cases: either $s(r) = s(r')$ or $e(r) = e(r')$. In both cases, it suffices to examine their headway at the central node. Each route arrives at or departs from the central node at time equal to its time slot. Therefore their headway is a nonzero multiple of $t = \frac{T}{L}$, which is an upper bound for OPT.

In a general spider network, we consider two cases. For two intersecting routes that pass through the central node, we can use the same argumentation as above for star networks. For two intersecting routes that lie in the same leg, the proof is similar to the proof of Theorem 6.3 for chains since it can be shown that the same properties hold considering either the central node or the tip of a leg as the first node of the chain (possibly with an appropriate time shift). □

6.5 PMS in Ring Networks

In the case of ring networks, that is, networks which consist of a single cycle, we may assume that all trains travel in the same direction (clockwise, without loss of generality), for the same reasons as for chains. Nodes are labeled by picking one arbitrarily and labeling it 0, then labeling every other node $1, \ldots, n-1$ starting from the neighbor of node 0 in the clockwise direction. We define $\tau(i, j)$ as the time distance from node i to node j in the clockwise direction. We also define the ring perimeter C as the total time needed to travel around the ring.

For ring networks we can distinguish between two cases, depending on whether the ring perimeter C is an integer multiple of the period T or not. In the following two sections we will analyze these cases.

6.5.1 The Case $C \equiv_T 0$

Theorem 6.6. *An instance of* PMS *in a ring with* $C \equiv_T 0$ *admits a solution of headway at least* $\frac{T}{k}$ *if and only if the corresponding* PC *instance can be colored with at most* k *colors.*

Proof. First, assume we are given a coloring of the routes with at most k colors. We can produce the desired schedule by using Algorithm 10 for PMS in chains, starting from Step 2 and using k instead of L. Let r and r' be two intersecting routes; without loss of generality assume that $s(r')$ is closer to 0 than $s(r)$ in the clockwise direction. Because $C \equiv_T 0$, it can be shown that it suffices to check their headway on only one of their common segments, even if there are two such segments.

Following similar arguments as those in the proof of Theorem 6.3, it can be shown that the headway is:

$$\min\{(\text{timeslot}(r) - \text{timeslot}(r')) \bmod T, (\text{timeslot}(r') - \text{timeslot}(r)) \bmod T\} \geq \frac{T}{k} . \tag{6.5}$$

For the inverse direction, suppose we have a schedule for the PMS instance with headway at least $\frac{T}{k}$. We will show how to obtain a coloring with k colors for the corresponding PC instance. For each route r, let $\text{timeslot}(r) = (\text{stime}(r) - \tau(0, s(r))) \bmod T$. Assign to r the color $w - 1$, where w is the smallest integer such that $\text{timeslot}(r) < w \cdot \frac{T}{k}$. Since w ranges from 1 to k and for any two intersecting routes r and r' their timeslots differ by at least $\frac{T}{k}$, this is a valid coloring. □

Corollary 6.7. PMS *in rings is* NP*-hard.*

Proof. We give a reduction from the decision version of PC in rings to the decision version of PMS in rings. PC is known to be NP-hard in rings [39]. Suppose we are given an instance of PC in a ring with n nodes and a path set $\mathcal{P}$, asking if $\mathcal{P}$ is colorable with k colors. We construct an instance of PMS in a ring with n nodes, routes identical to the paths in $\mathcal{P}$, inter-station distances of one time unit and $T = n$, asking if it is possible to achieve an objective function value of $\frac{T}{k}$. Clearly, the corresponding PC instance for the PMS instance we produced is the original PC instance. Therefore Theorem 6.6 applies, implying that the original PC instance can be colored with k colors if and only if a solution of value $\frac{T}{k}$ can be achieved for the PMS instance. □

At a first glance Theorem 6.6 seems to imply that a ρ-approximation algorithm for PC would give a $\frac{1}{\rho}$-approximation algorithm for PMS. However, this is true only in the case that the optimal solution for the PMS instance divides exactly the period T. The following example, illustrated in Figure 6.2, shows that in general the approximation obtained is smaller than $\frac{1}{\rho}$.

Let us denote by OPT_{PMS} the value of an optimal solution of an instance of PMS and by OPT_{PC} the cost of an optimal solution of the corresponding PC instance. We will present an infinite family of PMS instances in rings with $C \equiv_T 0$ for which OPT_{PMS} is strictly greater than $\frac{T}{\text{OPT}_{\text{PC}}}$, and is in fact asymptotically equal to $\frac{T}{\text{OPT}_{\text{PC}-1}}$. In this case we cannot directly utilize the approximation guarantee provided by an algorithm for PC to achieve an equivalent (inverse) guarantee for PMS.

Consider a ring of $2n$ nodes with time distances between successive nodes equal to 1 and time period $T = 2n$. The instance also consists of a set of $2n - 1$ routes $r_1, \ldots, r_{2n-1}$ with $s(r_i) = i - 1, i = 1 \ldots 2n - 1$, routes $r_1, \ldots, r_{2n-2}$ traveling across two edges and route r_{2n-1} traveling across three edges. The maximum congestion of this instance is $L = 2$. It is not hard to see that we need at least 3 colors to solve the corresponding PC instance. We will present a solution that achieves a minimum headway of $n - 1$,

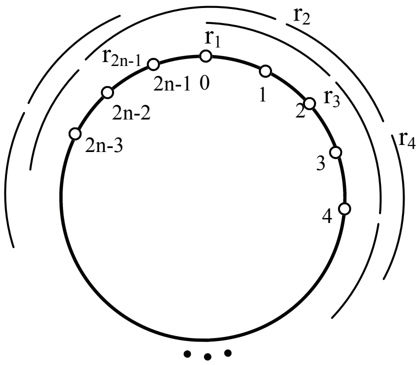

Figure 6.2: An infinite family of PMS instances in rings, in which a ρ-approximate solution for PC does not yield an $\frac{1}{\rho}$-approximate solution for PMS.

which is strictly greater than $\frac{T}{3} = \frac{2n}{3}$ for $n > 3$, and is asymptotically equal to $\frac{T}{2} = n$ for large n.

Assign to each even-numbered route r_{2i} starting time $\mathrm{stime}(r_{2i}) = i + \tau(0, s(r_{2i}))$ and to each odd-numbered route r_{2i+1} starting time $\mathrm{stime}(r_{2i+1}) = n + i + \tau(0, s(r_{2i+1}))$. The headway between an even-numbered route r_{2i} and the succeeding route r_{2i+1} is equal to n, while the headway between an odd-numbered route r_{2i+1} and the succeeding even-numbered route r_{2i+2} is equal to $n - 1$. Finally the headway between r_1 and r_{2n-1} is also $n - 1$. Therefore this solution achieves the desired headway.

The following theorem shows that it is still possible to use an approximate solution to PC in order to achieve an almost as good approximate solution for PMS.

Theorem 6.8. *A ρ-approximation algorithm for* PC *in rings implies an* $\left(\frac{1}{\rho}\frac{L}{L+1}\right)$-*approximation algorithm for* PMS *in rings with* $C \equiv_T 0$.

Proof. We will use the algorithm described in the proof of Theorem 6.6. We observe that $\mathrm{OPT}_{\mathrm{PMS}} < \frac{T}{\mathrm{OPT}_{\mathrm{PC}-1}}$ because a solution of PMS of value $\frac{T}{\mathrm{OPT}_{\mathrm{PC}-1}}$ would lead to a coloring with only $\mathrm{OPT}_{\mathrm{PC}} - 1$ colors by Theorem 6.6. Recall also that $\mathrm{OPT}_{\mathrm{PMS}} \leq \frac{T}{L}$.

A ρ-approximation algorithm for PC returns a solution $\mathrm{SOL}_{\mathrm{PC}} \leq \rho \cdot \mathrm{OPT}_{\mathrm{PC}}$. By Theorem 6.6 we can compute a solution for PMS of value $\mathrm{SOL}_{\mathrm{PMS}} = \frac{T}{\mathrm{SOL}_{\mathrm{PC}}} \geq \frac{1}{\rho} \cdot \frac{T}{\mathrm{OPT}_{\mathrm{PC}}}$. By the observations above it turns out that:

$$\mathrm{SOL}_{\mathrm{PMS}} \geq \frac{1}{\rho} \cdot \frac{T}{\frac{T}{\mathrm{OPT}_{\mathrm{PMS}}} + 1} = \frac{1}{\rho} \cdot \frac{T \cdot \mathrm{OPT}_{\mathrm{PMS}}}{T + \mathrm{OPT}_{\mathrm{PMS}}} \geq \frac{1}{\rho} \cdot \frac{L}{L + 1} \cdot \mathrm{OPT}_{\mathrm{PMS}} \ . \tag{6.6}$$

□

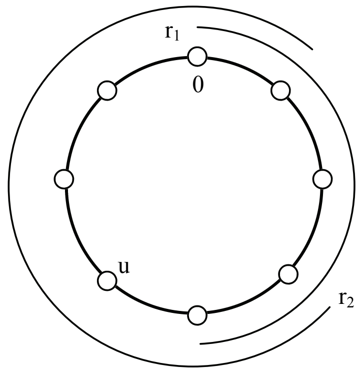

Figure 6.3: An example showing that the "path coloring" technique does not work for rings with $C \not\equiv_T 0$. Assuming $\tau(0, u) = T$ and $\tau(u, 0) = \frac{T}{2}$, the path coloring technique would assign time slots 0 and $\frac{T}{2}$ to routes r_1 and r_2 respectively and the two routes would collide at node 0 at any time which is an integer multiple of T.

Note that the family of instances presented above shows that the analysis of Theorem 6.8 is tight: the optimal solution is almost $\frac{T}{2} = n$ and the solution produced by directly exploiting an exact solution to PC is $\frac{2n}{3} = \frac{L}{L+1}n$.

Corollary 6.9. *There is a* $\left(\frac{2}{3} \cdot \frac{L}{L+1}\right)$*-approximation algorithm and a* $\left(0.73 \cdot \frac{L}{L+1}\right)$*-approximation randomized algorithm for* PMS *in rings with* $C \equiv_T 0$.

Proof. By using Theorem 6.8 and the deterministic approximation algorithm of Karapetian [45] and the randomized approximation algorithm of Kumar [48] that achieve ratios $\frac{3}{2}$ and 1.368, respectively. □

6.5.2 The Case $C \not\equiv_T 0$

Consider a ring network with n nodes and two routes r_1, r_2 with $0 = s(r_1) < e(r_2) < s(r_2) < e(r_1)$ and $\tau(s(r_1), s(r_2)) = x$. Let t_1, t_2 be the moments in time where the trains traveling along r_1 and r_2 arrive at node 0. These trains reach node $s(r_2)$ at times $(t_1 + x) \bmod T$ and $(t_2 - D + x) \bmod T$ respectively, where $D = C \bmod T$. As a result, in order to maximize the minimum headway between the two trains, we have to take into account the following time differences: $(t_1 - t_2) \bmod T$, $(t_2 - t_1) \bmod T$, $(t_1 - t_2 + D) \bmod T$ and $(t_2 - t_1 - D) \bmod T$. It is now clear that the algorithm of Theorem 6.6 may produce an infeasible solution if $D = (t_2 - t_1) \bmod T$ (see Figure 6.3). Therefore, we need a new algorithm for this case.

We propose Algorithm 12 for PMS in rings where $C \not\equiv_T 0$. In order to show the approximation ratio of Algorithm 12, we will need the following two lemmata.

Lemma 6.10. *Any two routes r, r' that belong to $\mathcal{P}_c$ have headway at least $\frac{T}{5L'}$. The same holds for any two routes that belong to $\mathcal{P}_0$.*

Proof. First, observe that any two routes in $\mathcal{P}_c$ have headway at least $\frac{T}{5L'}$ in a scheduling produced by the algorithm, because their arrangement is essentially the same as in the case of a chain network.

Consider now two routes r, r' in $\mathcal{P}_0$ to which the algorithm has assigned time slots timeslot(r) and timeslot(r'), respectively. Suppose, without loss of generality, that $s(r') > s(r)$. It is clear that since they are assigned different time slots these two routes cannot have a headway of less than $\frac{T}{5L'}$ at node 0 and therefore neither at node $s(r')$. We now need to show that their headway is not less than $\frac{T}{5L'}$ at node $s(r)$. We should examine two cases depending on whether $e(r) < e(r')$ or not.

Suppose $e(r') > s(r)$. In that case, $e(r') > e(r)$ and r' will be assigned a time slot before r. Route r' will reach $s(r)$ at time $\mathrm{stime}(r') + \tau(s(r'), 0) + \tau(0, s(r)) \equiv_T \mathrm{timeslot}(r') + \tau(0, s(r))$. Route r departs from $s(r)$ at time $\mathrm{stime}(r) = \mathrm{timeslot}(r) - \tau(s(r), 0) \equiv_T \mathrm{timeslot}(r) - D + \tau(0, s(r))$. However, the headway between $\mathrm{timeslot}(r') + D$ and timeslot(r) is at least $\frac{T}{5L'}$, because r' was assigned a time slot before r and $\mathrm{timeslot}(r') + D$ was excluded from S_0.

Let us now assume that $e(r') < s(r)$. In that case the two routes have only one common segment that starts at $s(r')$ and contains 0. Therefore, the fact that they have been assigned different time slots suffices to guarantee that their headway is at least $\frac{T}{5L'}$. □

Lemma 6.11. *Any two routes $r \in \mathcal{P}_c$ and $r' \in \mathcal{P}_0$ have headway at least $\frac{T}{5L'}$.*

Proof. We need to show that their headway is at least $\frac{T}{5L'}$ only at nodes $s(r)$ and $s(r')$ since these are the first nodes of the two possible common segments of r and r'.

Let t_0 = timeslot(r') be the time when r' passes through node 0. r' reaches $s(r)$ (if $s(r)$ is contained in r') at time $(\tau(0, s(r)) + t_0) \bmod T$, and, since $\mathrm{stime}(r) = (\tau(0, s(r)) + \mathrm{timeslot}(r)) \bmod T$, if r and r' had a headway of less than $\frac{T}{5L'}$ then they would have been assigned the same time slot, which is a contradiction.

Route r arrives at $s(r')$ at time $\mathrm{stime}(r) + \tau(s(r), s(r')) = \mathrm{timeslot}(r) + \tau(0, s(r'))$ while r' departs from $s(r')$ at time $\mathrm{stime}(r') = \mathrm{timeslot}(r') - \tau(s(r'), 0) \equiv_T \mathrm{timeslot}(r') - D + \tau(0, s(r'))$. If r and r' had headway of less than $\frac{T}{5L'}$, then

Algorithm 12 An algorithm for PMS in ring networks with $C \not\equiv_T 0$

Input: an instance $\langle G, t, T, \mathcal{R}\rangle$ of PMS, where G is a ring with $C \not\equiv_T 0$

Split $\mathcal{R}$ into two sets $\mathcal{P}_0$ and $\mathcal{P}_c$. $\mathcal{P}_0$ contains routes that pass through node 0 (i.e., having node 0 as an intermediate node) and $\mathcal{P}_c = \mathcal{R} \setminus \mathcal{P}_0$. Let $L_0 = |\mathcal{P}_0|$ and L_c be the maximum congestion with respect to $\mathcal{P}_c$.

Define $t = \frac{T}{5L'}$ and two sets of available time slots as follows: $S_0 = \{0, t, 2t, \ldots, (5L' - 1)t\}$, $S_c = \emptyset$ where $L' = \max\{L_0, L_c\}$.

Assign colors to routes of $\mathcal{P}_c$ by using an algorithm for PC in chains.

for all colors k, $1 \le k \le L_c$ **do**

    if $S_c \neq \emptyset$ **then**

        Select an item l from S_c.

    else

        Select an item l from S_0.

    end if

    Set timeslot(k) = l.

    for all routes r colored with color k **do**

        Assign departure time stime(r) = timeslot(k) + $\tau(0, s(r))$.

    end for

    Remove l from S_c and S_0.

    Move all time slots whose difference from $l+D$ is smaller than t from S_0 to S_c.

end for

Sort routes in $\mathcal{P}_0$ in non-increasing order of ending point.

for all $r \in \mathcal{P}_0$ in the sorted order **do**

    Select an item l from S_0.

    Set timeslot(r) = l.

    Set stime(r) = (timeslot(r) $- \tau(s(r), 0)$) mod T.

    Remove l from S_0.

    Remove from S_0 all time slots whose difference from $l + D$ is smaller than t.

end for

timeslot(r') would have a difference of less than $\frac{T}{5L'}$ from timeslot(r)+D which is also a contradiction, since all time slots which have a difference of less than $\frac{T}{5L'}$ from timeslot(r) + D were excluded from S_0 when r was assigned its time slot. □

Theorem 6.12. *Algorithm 12 is a $\frac{1}{5}$-approximation algorithm for* PMS *in rings with $C \not\equiv_T 0$.*

Proof. First, let us observe that $6L'$ time slots suffice to arrange the departure times of all routes. As far as routes in $\mathcal{P}_c$ are concerned, each one uses one time slot and excludes at most two others from S_0, in total using at most $3L_c$ time slots. Similarly, routes in $\mathcal{P}_0$ use at most $3L_0$ time slots.

A more precise analysis reveals that at most $2L_c$ time slots are used by Algorithm 12 to arrange routes in $\mathcal{P}_c$, provided that $L_c \geq 3$. Define for every color k used in the first phase of the algorithm cost(k) to be the number of items removed from S_0 for that color. If $S_c \neq \emptyset$ when color k is examined, then cost(k) ≤ 2. If $S_c = \emptyset$, this implies that cost(k) = 3, but cost(k') ≤ 1 for color k' which was examined immediately before k. Therefore the average cost for k and k' is at most 2. Notice that in the special case of the first color examined, the three time slots that must be excluded from S_0 may be used to accomodate the next two colors giving a total cost of 6 for the three colors. As a result, the average cost for all of the colors is at most 2, leading to at most $2L_c$ time slots being removed from S_0 for routes in $\mathcal{P}_c$.

The difference between time slots in Algorithm 12 is $\frac{T}{5L'}$. We only need to show that the headway between any two intersecting routes at any point is not smaller than the difference of the corresponding time slots. Since the algorithm assigns different time slots to intersecting routes, the minimum headway is at least $\frac{T}{5L'}$. The three possible cases for two routes r and r' of the ring have been examined in Lemmata 6.10 and 6.11. The proof is now complete. □

6.6 PMS in Tree Networks

In the case of tree networks one might attempt to use Algorithm 11 for spiders, after picking an arbitrary node 0. However, this idea may lead to the production of a solution with zero headway. Figure 6.4 illustrates this situation.

However, if we consider tree networks in which the time needed to travel along each edge is a multiple of $\frac{T}{2}$ it turns out that we can use a simple variation of Algorithm 10. In these networks the following useful property holds.

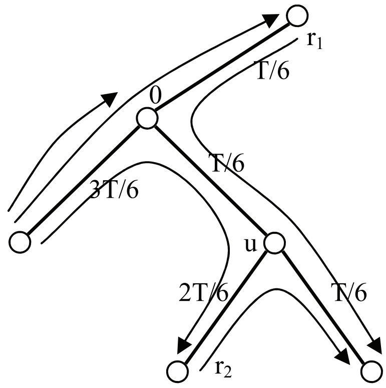

Figure 6.4: An example showing that Algorithm 11 for spiders does not work for trees. Assuming that route r_1 is assigned time slot 0 and route r_2 is assigned time slot $\frac{T}{3}$, route r_1 collides with route r_2 at node u at time $\frac{T}{6}$.

Remark 6.13. *For any three nodes a, b, c: $\tau(a, b) + \tau(b, c) \equiv_T \tau(a, c)$.*

Theorem 6.14. *An instance of* PMS *in a tree where the time needed to travel along each edge is a multiple of $\frac{T}{2}$ admits a solution of headway at least $\frac{T}{k}$ if and only if the corresponding* PC *instance can be colored with at most k colors.*

Proof. For the "if" direction we can produce the desired schedule by picking a node arbitrarily, labeling it 0, and then using Algorithm 10 for PMS in chains, starting from Step 2 and using k instead of L.

Assume there are two routes r and r' intersecting on a single edge $e = (u, v)$. Route r reaches node u at time $(\text{timeslot}(r) + \tau(0, s(r)) + \tau(s(r), u)) \bmod T = (\text{timeslot}(r) + \tau(0, u)) \bmod T$. By the same reasoning route r' reaches node u at time $(\text{timeslot}(r') + \tau(0, u)) \bmod T$. Hence, the headway of the two routes is equal to $(\text{timeslot}(r') - \text{timeslot}(r)) \bmod T$ which is clearly at least $\frac{T}{k}$.

For the "only if" direction, we pick an arbitrary node 0 and, for each route r, we consider the value $\text{timeslot}(r) = \text{stime}(r) - \tau(0, s(r))$. Following the proof of Theorem 6.6 and using Remark 6.13, we obtain a valid coloring with k colors for the original PC instance. □

Corollary 6.15. PMS *in trees is* NP*-hard.*

Proof. We will reduce the decision version of PC in trees to the decision version of PMS in trees. Given a PC instance and an integer k we will construct a PMS instance with time distances between nodes equal to one time unit and period $T = 2$. Theorem 6.14 implies that it is possible to achieve a solution of value at least $\frac{T}{k}$ if and only if the original PC instance can be colored with at most k colors. □

Theorem 6.16. *A ρ-approximation algorithm for* PC *in bidirectional trees implies a* $\left(\frac{1}{\rho}\frac{L}{L+1}\right)$*-approximation algorithm for* PMS *in trees where the time distances between nodes are multiples of* $\frac{T}{2}$.

Proof. The key observation is that if $\mathrm{OPT_{PMS}} \geq \frac{T}{\mathrm{OPT_{PC}}-1}$, then by using the algorithm of Theorem 6.14 we could achieve a coloring with $\mathrm{OPT_{PC}} - 1$ colors, which is a contradiction. Therefore $\mathrm{OPT_{PMS}} < \frac{T}{\mathrm{OPT_{PC}}-1}$ and the rest of the proof follows along the lines of the proof of Theorem 6.8. □

Corollary 6.17. *There is a* $\left(\frac{3}{5}\frac{L}{L+1}\right)$*-approximation algorithm for* PMS *in trees where the time distances between nodes are multiples of* $\frac{T}{2}$.

Proof. By using Theorem 6.16 and the $\frac{5}{3}$-approximation algorithm of Erlebach et al. [34]. □

Similarly to the case for rings with $C \equiv_T 0$, the $\frac{L}{L+1}$ factor can be justified by presenting an infinite family of instances having $\mathrm{OPT_{PMS}}$ strictly greater than $\frac{T}{\mathrm{OPT_{PC}}}$ and asymptotically equal to $\frac{T}{\mathrm{OPT_{PC}}-1}$. Consider a chain of $2n+2$ nodes numbered $0, \ldots, 2n+1$, with additional edges sticking out of nodes 1 and $2n$ connecting them to nodes v and v' respectively, a time period $T = 2n$ and all edges having time distance $\frac{T}{2} = n$. The instance consists of the following $2n+3$ routes: r_A from v' to v, r_B from 0 to v, r_C from v' to $2n+1$ and r_i, $i = 0, \ldots, 2n-1$ from i to $i+2$, resulting in a maximum congestion of $L = 2$.

The corresponding PC instance requires 3 colors but there is a PMS solution that achieves a headway of $n-1$, which is greater than $\frac{T}{3}$ for $n > 3$. Since the time distance of every edge is a multiple of $\frac{T}{2}$ we can pick an arbitrary root, assign a time slot to each route and set the starting time as the sum of the time slot and the distance of the starting node from the root. As shown above, this ensures that the headway between intersecting routes is the difference of their respective time slots.

Set $\mathrm{timeslot}(r_A) = 0$, $\mathrm{timeslot}(r_B) = \mathrm{timeslot}(r_C) = \frac{T}{2} = n$. For the even-numbered routes set $\mathrm{timeslot}(r_{2i}) = i$ and for the odd-numbered routes $\mathrm{timeslot}(r_{2i+1}) = \frac{T}{2} + i = n + i$. It is clear that this arrangement achieves a headway of $\frac{T}{2}$ between r_A,r_B and r_A,r_C, and a headway of at least $n-1 = \frac{T}{2}-1$ between successively numbered routes. In addition $\mathrm{timeslot}(r_{2n-1}) = 2n-1$ leading to a headway of $n-1$ between r_C and r_{2n-1} as well. Therefore, this is a solution which achieves a headway of $\frac{T}{2} - 1$. For an illustration see Figure 6.5.

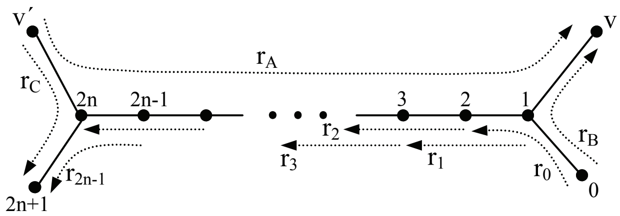

Figure 6.5: An infinite family of PMS instances in trees, in which a ρ-approximate solution for PC does not yield an $\frac{1}{\rho}$-approximate solution for PMS.

The infinite family of PMS instances presented shows that the analysis of Theorem 6.16 is tight: the optimal solution is almost n while the approximate solution produced by exploiting an exact coloring is $\frac{2n}{3} = \frac{L}{L+1}n$.

6.7 Conclusions

We have introduced the PERIODIC METRO SCHEDULING problem, which aims at generating a periodic timetable for a given set of routes and a given time period, in such a way that the minimum headway is maximized.

We have presented exact algorithms for chain and spider networks, and constant ratio approximation algorithms for ring networks, as well as for a special class of tree networks. Some of our algorithms make use of a reduction to PATH COLORING. We have left open the question of the approximability of PMS in general tree networks. Another interesting open question is to study the model where only the end stations of a route are given and one should determine both a path for each route and a departure time; this model applies to topologies that contain cycles, such as rings, grids and trees of rings.

Bibliography

[1] *INFOCOM 2006. 25th IEEE International Conference on Computer Communications, Joint Conference of the IEEE Computer and Communications Societies, 23-29 April 2006, Barcelona, Catalunya, Spain*, IEEE, 2006.

[2] Matthew Andrews and Lisa Zhang, *Wavelength assignment in optical networks with fixed fiber capacity*, ICALP (Josep Díaz, Juhani Karhumäki, Arto Lepistö, and Donald Sannella, eds.), Lecture Notes in Computer Science, vol. 3142, Springer, 2004, pp. 134–145.

[3] ______, *Complexity of wavelength assignment in optical network optimization*, in *INFOCOM* [1].

[4] ______, *Minimizing maximum fiber requirement in optical networks*, J. Comput. Syst. Sci. **72** (2006), no. 1, 118–131.

[5] Elliot Anshelevich, Anirban Dasgupta, Jon M. Kleinberg, Éva Tardos, Tom Wexler, and Tim Roughgarden, *The price of stability for network design with fair cost allocation*, FOCS, IEEE Computer Society, 2004, pp. 295–304.

[6] Baruch Awerbuch, Yossi Azar, Amos Fiat, Stefano Leonardi, and Adi Rosén, *On-line competitive algorithms for call admission in optical networks*, Algorithmica **31** (2001), no. 1, 29–43.

[7] Evangelos Bampas, Georgia Kaouri, Michael Lampis, and Aris Pagourtzis, *Periodic metro scheduling*, ATMOS (Riko Jacob and Matthias Müller-Hannemann, eds.), Dagstuhl Seminar Proceedings, vol. 06002, Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, 2006.

[8] Evangelos Bampas, Aris Pagourtzis, George Pierrakos, and Katerina Potika, *On a non-cooperative model for wavelength assignment in multifiber optical networks*, ISAAC (Seok-Hee Hong, Hiroshi Nagamochi,

and Takuro Fukunaga, eds.), Lecture Notes in Computer Science, vol. 5369, Springer, 2008, pp. 159–170.

[9] Evangelos Bampas, Aris Pagourtzis, George Pierrakos, and Vasileios Syrgkanis, *Colored resource allocation games*, CTW (Sonia Cafieri, Antonio Mucherino, Giacomo Nannicini, Fabien Tarissan, and Leo Liberti, eds.), 2009, pp. 68–72.

[10] Evangelos Bampas, Aris Pagourtzis, and Katerina Potika, *Maximum request satisfaction in WDM rings: Algorithms and experiments*, PCI (Theodore S. Papatheodorou, Dimitris N. Christodoulakis, and Nikitas N. Karanikolas, eds.), Current Trends in Informatics, vol. A, New Technologies Publications, 2007, pp. 627–642.

[11] ———, *Maximum profit wavelength assignment in WDM rings*, CTW, University of Milan, 2008, pp. 35–38.

[12] ———, *An experimental study of maximum profit wavelength assignment in WDM rings*, Networks (2009), to appear.

[13] Ron Banner and Ariel Orda, *Bottleneck routing games in communication networks*, in *INFOCOM* [1].

[14] Alan A. Bertossi, Paolo Carraresi, and Giorgio Gallo, *On some matching problems arising in vehicle scheduling models*, Networks **17** (1987), no. 3, 271–281.

[15] Vittorio Bilò, Michele Flammini, and Luca Moscardelli, *On Nash equilibria in non-cooperative all-optical networks*, STACS (Volker Diekert and Bruno Durand, eds.), Lecture Notes in Computer Science, vol. 3404, Springer, 2005, pp. 448–459.

[16] Vittorio Bilò and Luca Moscardelli, *The price of anarchy in all-optical networks*, SIROCCO (Rastislav Kralovic and Ondrej Sýkora, eds.), Lecture Notes in Computer Science, vol. 3104, Springer, 2004, pp. 13–22.

[17] Dietrich Braess, *Über ein Paradoxon aus der Verkehrsplanung*, Unternehmensforschung **12** (1968), 258–268, available in English in [18].

[18] Dietrich Braess, Anna Nagurney, and Tina Wakolbinger, *On a paradox of traffic planning*, Transport. Sci. **39** (2005), no. 4, 446–450.

[19] Costas Busch and Malik Magdon-Ismail, *Atomic routing games on maximum congestion*, AAIM (Siu-Wing Cheng and Chung Keung Poon, eds.), Lecture Notes in Computer Science, vol. 4041, Springer, 2006, pp. 79–91.

[20] Michael R. Bussieck, Thomas Winter, and Uwe Zimmermann, *Discrete optimization in public rail transport*, Math. Program. **79** (1997), 415–444.

[21] Ioannis Caragiannis, *Wavelength management in WDM rings to maximize the number of connections*, STACS (Wolfgang Thomas and Pascal Weil, eds.), Lecture Notes in Computer Science, vol. 4393, Springer, 2007, pp. 61–72.

[22] Martin C. Carlisle and Errol L. Lloyd, *On the k-coloring of intervals*, Discrete Appl. Math. **59** (1995), no. 3, 225–235.

[23] Steve Chien and Alistair Sinclair, *Convergence to approximate Nash equilibria in congestion games*, SODA (Nikhil Bansal, Kirk Pruhs, and Clifford Stein, eds.), SIAM, 2007, pp. 169–178.

[24] George Christodoulou and Elias Koutsoupias, *The price of anarchy of finite congestion games*, STOC (Harold N. Gabow and Ronald Fagin, eds.), ACM, 2005, pp. 67–73.

[25] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to algorithms*, second ed., MIT Press, Cambridge, MA, USA, 2001.

[26] Geir Dahl, *Disjoint congruence classes and a timetabling application*, Discrete Appl. Math. **157** (2009), 1702–1710.

[27] George B. Dantzig and Delbert R. Fulkerson, *Minimizing the number of tankers to meet a fixed schedule*, Nav. Res. Logist. Q. **1** (1954), no. 3, 217–222.

[28] Reinhard Diestel, *Graph theory*, third ed., Graduate Texts in Mathematics, vol. 173, Springer-Verlag Berlin Heidelberg, 2006.

[29] Rudra Dutta and George N. Rouskas, *A survey of virtual topology design algorithms for wavelength routed optical networks*, Optical Networks **1** (2000), no. 1, 73–89, also available as technical report TR-99-06, North Carolina State University, College of Engineering, Computer Science department.

[30] Stephan Eidenbenz, Aris Pagourtzis, and Peter Widmayer, *Flexible train rostering*, ISAAC (Toshihide Ibaraki, Naoki Katoh, and Hirotaka Ono, eds.), Lecture Notes in Computer Science, vol. 2906, Springer, 2003, pp. 615–624.

[31] Thomas Erlebach, Martin Gantenbein, Daniel Hürlimann, Gabriele Neyer, Aris Pagourtzis, Paolo Penna, Konrad Schlude, Kathleen Steinhöfel, David Scot Taylor, and Peter Widmayer, *On the complexity of train assignment problems*, ISAAC (Peter Eades and Tadao Takaoka, eds.), Lecture Notes in Computer Science, vol. 2223, Springer, 2001, pp. 390–402.

[32] Thomas Erlebach and Klaus Jansen, *Maximizing the number of connections in optical tree networks*, ISAAC (Kyung-Yong Chwa and Oscar H. Ibarra, eds.), Lecture Notes in Computer Science, vol. 1533, Springer, 1998, pp. 179–188.

[33] ______, *The maximum edge-disjoint paths problem in bidirected trees*, SIAM J. Discrete Math. **14** (2001), no. 3, 326–355.

[34] Thomas Erlebach, Klaus Jansen, Christos Kaklamanis, Milena Mihail, and Pino Persiano, *Optimal wavelength routing on directed fiber trees*, Theor. Comput. Sci. **221** (1999), no. 1-2, 119–137.

[35] Thomas Erlebach, Aris Pagourtzis, Katerina Potika, and Stamatis Stefanakos, *Resource allocation problems in multifiber WDM tree networks*, WG (Hans L. Bodlaender, ed.), Lecture Notes in Computer Science, vol. 2880, Springer, 2003, pp. 218–229.

[36] Eyal Even-Dar, Alexander Kesselman, and Yishay Mansour, *Convergence time to Nash equilibrium in load balancing*, ACM Trans. Alg. **3** (2007), no. 3.

[37] Dimitris Fotakis, Spyros C. Kontogiannis, Elias Koutsoupias, Marios Mavronicolas, and Paul G. Spirakis, *The structure and complexity of Nash equilibria for a selfish routing game*, ICALP (Peter Widmayer, Francisco Triguero Ruiz, Rafael Morales Bueno, Matthew Hennessy, Stephan Eidenbenz, and Ricardo Conejo, eds.), Lecture Notes in Computer Science, vol. 2380, Springer, 2002, pp. 123–134.

[38] Michael R. Garey and David S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, W.H. Freeman and Company, 1979.

[39] Michael R. Garey, David S. Johnson, Gary L. Miller, and Christos H. Papadimitriou, *The complexity of coloring circular arcs and chords*, SIAM J. Alg. Disc. Meth. **1** (1980), no. 2, 216–227.

[40] Michael Gatto, Björn Glaus, Riko Jacob, Leon Peeters, and Peter Widmayer, *Railway delay management: Exploring its algorithmic complexity*, SWAT (Torben Hagerup and Jyrki Katajainen, eds.), Lecture Notes in Computer Science, vol. 3111, Springer, 2004, pp. 199–211.

[41] Michael Gatto, Riko Jacob, Leon Peeters, and Anita Schöbel, *The computational complexity of delay management*, WG (Dieter Kratsch, ed.), Lecture Notes in Computer Science, vol. 3787, Springer, 2005, pp. 227–238.

[42] Fanica Gavril, *Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph*, SIAM J. Comput. **1** (1972), no. 2, 180–187.

[43] Zülfükar Genç, *Ein neuer Ansatz zur Fahrplanoptimierung im ÖPNV: Maximierung von zeitlichen Sicherheitabständen*, Ph.D. thesis, Universität zu Köln, Mathematisch-Naturwissenschaftliche Fakultät, Institut für Informatik, 2003, in German.

[44] George F. Georgakopoulos, Dimitris J. Kavvadias, and Leonidas G. Sioutis, *Nash equilibria in all-optical networks*, WINE (Xiaotie Deng and Yinyu Ye, eds.), Lecture Notes in Computer Science, vol. 3828, Springer, 2005, pp. 1033–1045.

[45] Iskandar A. Karapetian, *On the coloring of circular arc graphs*, Dokl. Akad. Nauk Armyan SSR **70** (1980), no. 5, 306–311, in Russian.

[46] Elias Koutsoupias and Christos H. Papadimitriou, *Worst-case equilibria*, STACS (Christoph Meinel and Sophie Tison, eds.), Lecture Notes in Computer Science, vol. 1563, Springer, 1999, pp. 404–413.

[47] Leo G. Kroon and Leon W.P. Peeters, *A variable trip time model for cyclic railway timetabling*, Transport. Sci. **37** (2003), no. 2, 198–212.

[48] Vijay Kumar, *Approximating circular arc colouring and bandwidth allocation in all-optical ring networks*, APPROX (Klaus Jansen and Dorit S. Hochbaum, eds.), Lecture Notes in Computer Science, vol. 1444, Springer, 1998, pp. 147–158.

[49] Guangzhi Li and Rahul Simha, *On the wavelength assignment problem in multifiber WDM star and ring networks*, IEEE/ACM Trans. Netw. **9** (2001), no. 1, 60–68.

[50] Christian Liebchen, *A cut-based heuristic to produce almost feasible periodic railway timetables*, WEA (Sotiris E. Nikoletseas, ed.), Lecture Notes in Computer Science, vol. 3503, Springer, 2005, pp. 354–366.

[51] Christian Liebchen and Rolf H. Möhring, *A case study in periodic timetabling*, Electr. Notes Theor. Comput. Sci. **66** (2002), no. 6.

[52] Tze-Heng Ma and Jeremy Spinrad, *Avoiding matrix multiplication*, WG (Rolf H. Möhring, ed.), Lecture Notes in Computer Science, vol. 484, Springer, 1990, pp. 61–71.

[53] Luciano Margara and Janos Simon, *Wavelength assignment problem on all-optical networks with k fibres per link*, ICALP (Ugo Montanari, José D. P. Rolim, and Emo Welzl, eds.), Lecture Notes in Computer Science, vol. 1853, Springer, 2000, pp. 768–779.

[54] Marios Mavronicolas and Paul G. Spirakis, *The price of selfish routing*, STOC, 2001, pp. 510–519.

[55] Milena Mihail, Christos Kaklamanis, and Satish Rao, *Efficient access to optical bandwidth - wavelength routing on directed fiber trees, rings, and trees of rings*, FOCS, 1995, pp. 548–557.

[56] Igal Milchtaich, *Congestion games with player-specific payoff functions*, Games Econ. Behav. **13** (1996), no. 1, 111–124.

[57] Ioannis Milis, Aris Pagourtzis, and Katerina Potika, *Selfish routing and path coloring in all-optical networks*, CAAN (Jeannette C. M. Janssen and Pawel Pralat, eds.), Lecture Notes in Computer Science, vol. 4852, Springer, 2007, pp. 71–84.

[58] Dov Monderer and Lloyd S. Shapley, *Potential games*, Games Econ. Behav. **14** (1996), 124–143.

[59] Clyde L. Monma and Victor K.-W. Wei, *Intersection graphs of paths in a tree*, J. Comb. Theory B **41** (1986), no. 2, 141–181.

[60] John Nash, *Non-cooperative games*, Ann. Math. **54** (1951), no. 2, 286–295.

[61] Christos Nomikos, Aris Pagourtzis, Katerina Potika, and Stathis Zachos, *Routing and wavelength assignment in multifiber WDM networks with non-uniform fiber cost*, Computer Networks **50** (2006), no. 1, 1–14.

[62] Christos Nomikos, Aris Pagourtzis, and Stathis Zachos, *Routing and path multicoloring*, Inform. Process. Lett. **80** (2001), no. 5, 249–256.

[63] ———, *Minimizing request blocking in all-optical rings*, INFOCOM, 2003.

[64] ———, *Satisfying a maximum number of pre-routed requests in all-optical rings*, Comput. Netw. **42** (2003), no. 1, 55–63.

[65] Christos Nomikos and Stathis Zachos, *Coloring a maximum number of paths in a graph*, Workshop on Algorithmic Aspects of Communication, Bologna, July 1997.

[66] Katerina Potika, *Maximizing the number of connections in multifiber WDM chain, ring and star networks*, NETWORKING (Raouf Boutaba, Kevin C. Almeroth, Ramón Puigjaner, Sherman X. Shen, and James P. Black, eds.), Lecture Notes in Computer Science, vol. 3462, Springer, 2005, pp. 1465–1470.

[67] Robert W. Rosenthal, *A class of games possessing pure-strategy Nash equilibria*, Int. J. Game Theory **2** (1973), 65–67.

[68] Tim Roughgarden and Éva Tardos, *How bad is selfish routing?*, J. ACM **49** (2002), no. 2, 236–259.

[69] M. Saad and Zhi-Quan Luo, *On the routing and wavelength assignment in multifiber WDM networks*, IEEE J. Sel. Area. Comm. **22** (2004), no. 9, 1708–1717.

[70] Anita Schöbel, *A model for the delay management problem based on mixed-integer-programming*, Electr. Notes Theor. Comput. Sci. **50** (2001), no. 1.

[71] Alexander Schrijver, *Routing and timetabling by topological search*, Documenta Mathematica Extra Volume ICM III, 1998, pp. 687–695.

[72] Paolo Serafini and Walter Ukovich, *A mathematical model for periodic scheduling problems*, SIAM J. Discrete Math. **2** (1989), no. 4, 550–581.

[73] Vijay V. Vazirani, *Approximation algorithms*, Springer-Verlag Berlin Heidelberg, 2001.

[74] Adrian Vetta, *Nash equilibria in competitive societies, with applications to facility location, traffic routing and auctions*, FOCS, IEEE Computer Society, 2002, pp. 416–425.

[75] Peng-Jun Wan and Liwu Liu, *Maximal throughput in wavelength-routed optical networks*, Multichannel Optical Networks: Theory and Practice (Peng-Jun Wan, Ding-Zhu Du, and Panos M. Pardalos, eds.), DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 46, AMS, 1998, pp. 15–26.

[76] Peter Winkler and Lisa Zhang, *Wavelength assignment and generalized interval graph coloring*, SODA, 2003, pp. 830–831.

Thesis proudly powered by $\LaTeX$