Autoreferat

do wniosku z dnia 21 kwietnia 2015
o przeprowadzenie postępowania habilitacyjnego
w dziedzinie nauk matematycznych w dyscyplinie informatyka

I. Imię i Nazwisko: Grzegorz Stachowiak

II. Posiada stopień/tytuł:

1. doktor nauk matematycznych w dziedzinie informatyki – praca doktorska pt. *Generowanie wybranych klas obiektów kombinatorycznych algorytmami minimalnych zmian* (Wydział Matematyki, Fizyki i Chemii, Uniwersytet Wrocławski 1995).

2. magister matematyki, spec. teoretyczna (Wydział Matematyki, Fizyki i Chemii, Uniwersytet Wrocławski 1988)

III. Zatrudniony w: Instytut Informatyki Uniwersytetu Wrocławskiego (od 1 października 1988 do chwili obecnej). Do września 1996 jako asystent i od tego czasu jako adiunkt.

IV. Wskazanie osiągnięcia wynikającego z art. 16 ust. 2 ustawy z dnia 14 marca 2003 r. o stopniach naukowych i tytule naukowym oraz o stopniach i tytule w zakresie sztuki (Dz. U. nr 65, poz. 595 ze zm.):
Tytuł osiągnięcia: Sieci poprawiające – dokładne ograniczenia górne i dolne.

1. Grzegorz Stachowiak: Fibonacci Correction Networks. SWAT 2000, LNCS 1851: 535-548 (JCR)

2. Grzegorz Stachowiak: Fast Periodic Correction Networks. FCT 2003, LNCS 2751: 144-156 (JCR)

3. Grzegorz Stachowiak: Lower Bounds on Correction Networks. ISAAC 2003, LNCS 2906: 221-229 (JCR)

4. Grzegorz Stachowiak: Fast periodic correction networks. Theor. Comput. Sci. 354(3): 354-366 (2006) (JCR)

V. Pozostałe osiągnięcia i dorobek naukowy:

Prace wchodzące w skład rozprawy doktorskiej

5. Grzegorz Stachowiak: The Number of Linear Extensions of Bipartite Graphs, Order 5 (1988), 257 - 259. (JCR)

6. Grzegorz Stachowiak: A Relation between the Comparability Graph and the Number of Linear Extensions, Order 6 (1989), 241 - 244. (JCR)

7. Grzegorz Stachowiak: Hamilton Paths in Graphs of Linear Extensions for Unions of Posets. SIAM J. Discrete Math. 5(2): 199-206 (1992) (JCR)

8. Grzegorz Stachowiak: On a long cycle in the graph of all linear extensions of a poset consisting of two disjoint chains. Discrete Mathematics 131(1-3): 375-378 (1994) (JCR)

9. Grzegorz Stachowiak: Finding parity difference by involutions. Discrete Mathematics 163(1-3): 139-151 (1997) (JCR)

Pozostałe prace:

10. Marcin Kik, Miroslaw Kutylowski, Grzegorz Stachowiak: Periodic Constant Depth Sorting Networks. STACS 1994, LNCS 775: 201-212. (JCR)
11. Tomasz Jurdzinski, Grzegorz Stachowiak: Probabilistic Algorithms for the Wakeup Problem in Single-Hop Radio Networks. ISAAC 2002, LNCS 2518: 535-549. (JCR)
12. Tomasz Jurdzinski, Grzegorz Stachowiak: Probabilistic Algorithms for the Wake-Up Problem in Single-Hop Radio Networks. Theory Comput. Syst. 38(3): 347-367 (2005). (JCR)
13. Marcin Bienkowski, Marek Chrobak, Christoph Durr, Mathilde Hurand, Artur Jez, Lukasz Jez, Grzegorz Stachowiak: Collecting Weighted Items from a Dynamic Queue, SODA 2009, 1126-1135.
14. Grzegorz Stachowiak: Asynchronous Deterministic Rendezvous on the Line, SOFSEM 2009, LNCS 5404, 497-508.
15. Tomasz Jurdzinski, Dariusz R. Kowalski, Michal Rozanski, Grzegorz Stachowiak: On the impact of geometry on ad hoc communication in wireless networks. PODC 2014: 357-366
16. Marcin Bienkowski, Marek Chrobak, Christoph Dürr, Mathilde Hurand, Artur Jez, Lukasz Jez, Grzegorz Stachowiak: Collecting Weighted Items from a Dynamic Queue. Algorithmica 65(1): 60-94 (2013). (JCR)
17. Marcin Bienkowski, Marek Chrobak, Christoph Dürr, Mathilde Hurand, Artur Jez, Lukasz Jez, Grzegorz Stachowiak: A ϕ-competitive algorithm for collecting items with increasing weights from a dynamic queue. Theor. Comput. Sci. 475: 92-102 (2013). (JCR)
18. Tomasz Jurdzinski, Dariusz R. Kowalski, Grzegorz Stachowiak: Distributed Deterministic Broadcasting in Uniform-Power Ad Hoc Wireless Networks. FCT 2013, LNCS 8070: 195-209
19. Tomasz Jurdzinski, Dariusz R. Kowalski, Grzegorz Stachowiak: Distributed Deterministic Broadcasting in Wireless Networks of Weak Devices. ICALP 2013, LNCS 7966: 632-644
20. Marcin Bienkowski, Jaroslaw Byrka, Marek Chrobak, Lukasz Jez, Jiri Sgall, Grzegorz Stachowiak: Online Control Message Aggregation in Chain Networks. WADS 2013, LNCS 8037: 133-145
21. Tomasz Jurdzinski, Dariusz R. Kowalski, Michal Rozanski, Grzegorz Stachowiak: Distributed Randomized Broadcasting in Wireless Networks under the SINR Model. DISC 2013, LNCS 8205: 373-387

Opis wyników uzyskanych w ramach osiągnięcia naukowego:

Sortujące sieci komparatorów są klasycznym modelem obliczeń rozważanym w informatyce. Są one opisane w klasycznych podręcznikach akademickich takich jak [27, 24]. Głównym osiągnięciem habilitacyjnym jest konstrukcja sieci, które sortują dane wejściowe różniące się od ciągów posortowanych niewielką liczbą zaburzeń. Podstawowym zastosowaniem takich sieci jest rola „łatki", która dodana do zwykłej sieci sortującej, powoduje że całość jest odporna na stałą liczbę

pasywnych błędów. Pokazałem też, że głębokość czyli szybkość działania skonstruowanej przeze mnie „łatki" jest optymalna z dokładnością do addytywnego składnika niższego rzędu.

Sieć komparatorów składa się z n rejestrów $r_1, r_2, \ldots, r_n$, z których każdy zawiera jedną z n liczb. Zadaniem wykonywanym przez sieć jest posortowanie tych liczb czyli ustawienie w porządku od najmniejszej do największej. Operacje na zawartościach rejestrów wykonują podstawowe elementy zwane *komparatorami*. Komparator $[i : j]$ porównuje liczby w dwóch rejestrach r_i i r_j i zamienia je jeśli $i < j$ oraz liczba w r_j jest mniejsza. Komparatory rozmieszczone są w sieci przed rozpoczęciem sortowania i wyniki wcześniejszych porównań nie wpływają na położenie komparatorów później wykonujących obliczenia.

Komparatory łączące rozłączne pary rejestrów mogą wykonywać swoje operacje jednocześnie, co pozwala wydatnie zmniejszyć czas sortowania. Sieć komparatorów jest więc podzielona na podzbiory rozłącznych komparatorów zwane *warstwami*. Czas działania sieci komparatorów jest proporcjonalny do liczby warstw zwanej *głębokością* sieci. Koszt działania sieci komparatorów mierzy się dwoma parametrami: jej głębokością oraz *rozmiarem* czyli liczbą komparatorów.

Implementacje sieci komparatorów znajdują zastosowanie jako układy do sprzętowego sortowania i routingu. Najbardziej znaną siecią sortującą jest sieć Batchera [23] z 1968 roku o głębokości $\frac{1}{2}\log^2 n$. Jest to konstrukcja, która dla praktycznych rozmiarów n wejścia daje najmniejsze głębokości i rozmiary sieci sortującej. Inną praktyczną konstrukcję sieci sortującej o głębokości $O(\log^2 n)$ choć z nieco gorszą stałą ukrytą w notacji $O(\cdot)$ można otrzymać implementując sortowanie Shella.

Sieci te nie są jednak asymptotycznie optymalne — taka optymalna sieć została skonstruowana w 1983 roku prze Ajtaia Komlosa i Szemerediego [22]. Sieć ta ma głębokość $O(\log n)$ jednak ani w wersji oryginalnej, ani po ulepszeniach wprowadzonych przez późniejszych autorów stała ukryta za $O(\cdot)$ nie osiągnęła wartości przy których mogłaby ona konkurować z siecią Batchera dla praktycznych wartości n. Najmniejsza wartość tej stałej została uzyskana przez Chvatala i wynosi 1830.

Podstawowym narzędziem służącym do badania sieci komparatorów jest zasada zero-jedynkowa:

Fact 1 (zasada zero-jedynkowa) *Sieć komparatorów jest siecią sortującą wtedy i tylko wtedy gdy sortuje dowolny ciąg wejściowy złożony z zer i jedynek.*

W pracach wchodzących w skład osiągnięcia habilitacyjnego zajmuję się sieciami komparatorów sortującymi ciągi t-zaburzone. Ciąg *t-zaburzony* to taki, który różni się od posortowanego co najwyżej t transpozycjami par elementów. Alternatywnie można też powiedzieć, że ciąg zero-jedynkowy $x_1, x_2, \ldots, x_n$ jest t-zaburzony wtedy i tylko wtedy, gdy istnieje w nim indeks i, taki że co najwyżej t elementów spośród $x_1, x_2, \ldots, x_i$ jest jedynkami i co najwyżej t elementów spośród $x_{i+1}, \ldots, x_n$ jest zerami.

Problem sortowania takich ciągów bierze się z rozważań zmierzających do konstrukcji sieci komparatorów odpornych na błędy. W rozważaniach tych wadliwy komparator zamiast sortować zwraca liczby z wejścia w jakimkolwiek porządku. Zachodzi bowiem następujący fakt:

Spostrzeżenie 1 *Jeśli ciąg zero-jedynkowy przejdzie przez jakąkolwiek sieć sortującą z (co najwyżej) t błędami, to na wyjściu otrzymamy zero-jedynkowy ciąg t-zaburzony.*

Autorzy poprzednich prac dotyczących odporności sieci sortujących na błędy stosowali dwa podejścia do ich konstrukcji:

- Skonstruować sieć sortującą specjalnie z myślą o tym, żeby była odporna na błędy.
- Dołożyć do nieodpornej na błędy sieci sortującej dodatkową sieć komparatorów, która będzie „łatką" poprawiającą błędy obecne w tej sieci.

Należy również stwierdzić, że autorzy nie są zgodni co do tego jakiego rodzaju błędy mogą wystąpić w sieci i jak zdefiniowana odporność na błędy ich interesuje.

Rozważane rodzaje błędów to *błędy pasywne* przy których komparator nic nie robi zamiast wykonywać sortowanie pary liczb, *błędy odwracające* gdy wadliwy komparator zwraca elementy wejściowe odwrotnie uporządkowane i *błędy destrukcyjne* gdy wadliwy komparator może zwrócić dowolne liczby. Rozważa się dwa możliwe cele które chcemy osiągnąć przy konstrukcji sieci sortującej odpornej na błędy. Pierwszy to odporność na błędy losowe definiowana jako odporność całej sieci z prawdopodobieństwem $1 - 1/n$ na niezależne awarie wszystkich komparatorów zdarzające się ze stałym niewielkim prawdopodobieństwem. Drugi możliwy cel to odporność sieci na maksymalnie t błędów, które mogą zostać złośliwie rozłożone po sieci.

Odporność na błędy pasywne przy błędach losowych lub maksymalnie t błędach była rozważana przez Yao i Yao. Stwierdzili oni, że odporność na błędy losowe może być uzyskana dla AKS lub sieci Batchera poprzez powtórzenie każdego komparatora $c \log n$ razy. Odporność na losowe błędy destrukcyjne była po raz pierwszy rozważana przez Assafa i Upfala. Wymaga ona tworzenia wielu kopii sortowanych elementów a więc wyjścia poza tradycyjny model sieci komparatorów. Później problemem odporności na błędy losowe zajmowali się m.in. Leighton i Ma pokazując, że poprzez modyfikację sieci AKS można zbudować sieć odporną na błędy losowe o głębokości $O(\log n)$. Ich wynik ma wartość jedynie teoretyczną ponieważ skonstruowana w ten sposób sieć dziedziczy po AKS olbrzymią stałą ukrytą w notacji $O(\cdot)$.

W pracach koncentrujemy się na odporności na maksymalnie t błędów pasywnych. Odporność ta jest uzyskiwana przez dodanie do sieci sortującej drugiej sieci która sortuje ciągi t-zaburzone. Yao i Yao wskazali sieć o $2nt$ komparatorach która dodana do dowolnej sieci sortującej tworzy z nią sieć sortującą odporną na t błędów. Rozmiar jej jest asymptotycznie optymalny, ale duża głębokość $2nt$ czyni tę sieć mało użyteczną w praktyce. Problem konstrukcji takiej sieci przy $t = 1$ był rozważany przez Schimmlera i Starkego. Ich sieć miała głębokość $2 \log n$ i rozmiar $3.5 \log n$ co czyni ją atrakcyjną do zastosowań praktycznych. Nie rozwiązali oni jednak zadowalająco problemu dla $t > 1$ błędów proponując jedynie proste złożenie t sieci dla jednego błędu w celu poradzenia sobie z t błędami. Rozwiązanie to daje łączną głębokość $O(t \log n)$ i ma sens tylko dla małych t. Dopiero Marek Piotrów wpadł na pomysł, aby przekształcić sieć Schimmlera-Starkego tak by ustawiając jedną taką sieć za drugą można było je w siebie „wsunąć" uzyskując w ten sposób efekt pipeliningu. W ten sposób otrzymał on sieć o głębokości $O(\log n + t)$. Celem jego pracy nie było śrubowanie stałych stojących przy $\log n$ i t w symbolu $O(\cdot)$ i stałe te w pracy nie są policzone. Stała przy $\log n$ została oszacowana przez autora tego autoreferatu i wynosi co najmniej 22. W efekcie wynik z tej pracy jest czysto teoretyczny jakkolwiek praca ta może stanowić dobry punkt wyjścia do konstrukcji praktycznych sieci–łatek.

Godnym uwagi jest fakt, że mimo ukazania się tylu znakomitych prac nie wiadomo było cały czas jak głęboka musi być sieć komparatorów która sortuje każde zero-jedynkowe wejście które powstaje z posortowanego poprzez zamianę pojedynczego zera na jedynkę. Sieć Schimmlera i Starkego ma głębokość $2 \log n$. Pierwszym zaskakującym wynikiem w niniejszym osiągnięciu jest podanie konstrukcji sieci o głębokości $1.44 \log n + 2.81$ (można ją łatwo polepszyć do $1.44 \log n + 1.81$) opisanej w pracy [1].

Ograniczenie dolne $\log n$ na głębokość takiej sieci wynika z ograniczenia dolnego na problem selekcji w sieciach komparatorów. Pojawiło się więc pytanie, czy głębokość $1.44 \log n$ jest optymalna, czy można lepiej? Wobec niepowodzenia prób konstrukcji sieci o mniejszej głębokości podjęto próby wykazania, że jest to granica dolna co ostatecznie zrobiłem w pracy [3].

W pracach poprzednich autorów „łatki" zapewniające odporność na $t = 2, 3, 4$ błędy miały znacznie większą głębokość, niż dla $t = 1$. Kolejnym ważnym problemem była więc konstrukcja jak najlepszych sieci dodatkowych dla niewielkich stałych t. W pracy [1] udało mi się skonstruować sieci dla $t = 2, 3, 4$ o głębokości większej tylko o $O\left(\sqrt{\log n}\right)$ warstw niż dla $t = 1$,

czyli prawie o głębokości $1.44 \log n$. Wszystkie podane konstrukcje są praktyczne. W pracy tej dla dużych t uzyskałem dla dowolnego $\varepsilon > 0$ sieci o głębokości $(1.44 + \varepsilon) \log n + O(t)$. Sieci te mają asymptotycznie optymalny rozmiar $O(nt)$. Na uwagę zasługuje, że sieci skonstruowane w pracy [1] do poprawiania t błędów, dla praktycznych n i t mają mniejszą głębokość i rozmiar, niż najlepsze dotąd pod tym względem sieci Schimmlera i Starkego.

Innym problemem jest, niekoniecznie odporne na błędy, sortowanie ciągów t-zaburzonych. Problem ten można spotkać w wielu zastosowaniach praktycznych. Dla $t = 1$ jest on równoważny problemowi poprzednio rozważanemu. Dla wyższych t jest słabszą jego wersją, a zatem zapewne można go rozwiązać mniejszym kosztem. Sieć sortującą wszystkie ciągi t zaburzone nazywamy t-poprawiającą. Kik, Kutyłowski i Piotrów [25] podali konstrukcję sieci t-poprawiającej o głębokości $4 \log n + O\left((\log \log n \log t)^2\right)$. W pracy [1] podajemy konstrukcję sieci t-poprawiającej o głębokości

$$(1.44 + \varepsilon) \log n + O(\log \log n \log t).$$

Sieć ta ma optymalny asymptotycznie rozmiar $O(n \log t)$.

Ostatni rozważany przez nas problemem wiąże się z okresowymi sieciami sortującymi. Okresowa sieć komparatorów to taka, która jest stosowana nie raz tylko wiele razy do posortowania danego wejścia. Dodatkowym parametrem, poza rozmiarem i głębokością, określającym jakość sieci jest *liczba iteracji* czyli to, ile razy trzeba zastosować sieć, żeby być pewnym, że ciąg jest posortowany. Sieciom takim poświęcona była m. in. praca [10]. Najlepsza znana okresowa sieć sortująca głębokości 3 o liczbie iteracji $O(\log^3 n)$ była skonstruowana przez Lorysia et al. [28].

Problem konstrukcji sieci okresowych, których zadaniem jest jedynie sortowanie ciągów t-zaburzonych był rozważany przez poprzednich autorów [26, 30]. Skonstruowali oni okresowe sieci poprawiające głębokości 8 i 6 o liczbie iteracji $O(\log n + t)$. Analiza tych sieci opiera się na fakcie, że gdy mamy ciąg zaburzony z t „zagubionymi" jedynkami, to pierwsza z tych jedynek przestaje być zagubiona po czasie $O(\log n)$, a każda następna potrzebuje na to dodatkowo $O(1)$ kroków. Problemem jednak było skonstruowanie jakiejkolwiek sieci o stałej głębokości i liczbie iteracji $O(\log n + o(t))$. Problem ten rozwiązałem w pracy [2], gdzie opisana jest okresowa sieć poprawiająca głębokości 3 o liczbie iteracji $O(\log n + \operatorname{polylog} t)$ i wymagało to zastosowania zupełnie innej analizy działania sieci.

W związku z wyczerpaniem się tematyki sieci komparatorów w późniejszych pracach zajmowałem się innymi zagadnieniami, w większości również dotyczącymi algorytmów rozproszonych.

Opis pozostałych osiągnięć:

Prace przed doktoratem. Przed doktoratem zajmowałem się problemami zliczania i generowania rozszerzeń liniowych zbiorów częściowo uporządkowanych. W pracach [5, 6] zajmowałem się liczbami rozszerzeń liniowych posetów, które powstają przez acykliczne zorientowanie krawędzi grafu porównywalności. Pokazałem w nich, że największą liczbę rozszerzeń liniowych dostajemy gdy po zorientowaniu otrzymamy poset dla którego ten graf jest grafem porównywalności — w pierwszej pracy w szczególnym przypadku grafów dwudzielnych, a w drugiej w ogólnym przypadku. W drugiej pracy zdefiniowałem też funkcję dla grafów o wartościach naturalnych, która dla grafu porównywalności równa jest liczbie rozszerzeń liniowych stowarzyszonego z nim posetu.

Praca [7] zawiera jeden z najciekawszych wyników tej serii artykułów. Jest to jedna z najczęściej cytowanych prac z tego autoreferatu min. w czwartym tomie *The Art of Computer Programming* Donalda Knutha. Pokazałem w niej, że jeśli poset ma parzystą liczbę rozszerzeń liniowych, które mogą być generowane przez sąsiednie transpozycje, to rozszerzenia liniowe sumy

tego i dowolnego innego posetu mogą być również generowane przez sąsiednie transpozycje. Jednym z wniosków z tego wyniku jest twierdzenie, że permutacje multisetu mogą być generowane przez sąsiednie transpozycje gdy dwa rodzaje elementów występują w nieparzystej liczbie kopii. W pracy [8] pokazałem, że ciągi zawierające k zer i l jedynek mogą być generowane różne od ciągów $0^k1^l, 1^l0^k$ mogą być generowane przez sąsiednie transpozycje tak by pierwszy i ostatni generowany ciąg sąsiadowały (było to rozwiązanie otwartego problemu F. Ruskey'a).

W ramach doktoratu zajmowałem się też w pracy [9] problemami obliczania *różnicy parzystości* dla zbiorów częściowo uporządkowanych. Jeśli rozszerzenia liniowe można generować przez transpozycje, to różnica ta powinna być równa 0 lub 1. Pokazałem sposoby obliczania różnicy parzystości dla drzew za pomocą inwolucji. Pokazałem też, że obliczanie tej różnicy dla dowolnych posetów jest $\#P$-trudne. Niezależnie od tego w nieopublikowanej pracy pokazałem, że różnica parzystości jest niezmiennikiem porównywalności.

Przed doktoratem opublikowana też była pierwsza z prac na temat sieci komparatorów *Periodic constant depth sorting network* [10]. Dalsze prace z sieci komparatorów opisane są jako osiągnięcie habilitacyjne. Mimo że znajduje się ona w tomie LNCS sprzed 2006 roku, który jest na liście filadelfijskiej, to nie ma go w bazie WoS. Zatem nie mogłem jej wliczyć do moich statystyk cytowalności mimo dużej liczby cytowań w Google Scholar lub Citeseer. Praca ta poświęcona jest konstrukcji okresowej sieci o stałej głębokości, która w ciągu $O(n^\varepsilon)$ iteracji sortuje dowolne n elementów. Stała głębokość rośnie odwrotnie proporcjonalnie do spadku parametru ε. Konstrukcja nie jest praktyczna w związku z tym, że opiera się na ε-halverach. Mój wkład polegał na wykonaniu analizy skonstruowanej okresowej sieci sortującej.

Asynchroniczne rendezvous. W pracy [14] dyskutowany jest problem spotkania się (rendezvous) dwóch robotów w grafie będącym nieskończoną ścieżką. Rozważany jest model asynchroniczny, w którym oba roboty mogą poruszać się z dowolnymi i dowolnie zmieniającymi się prędkościami. Roboty ustalają swoje trasy deterministycznie na podstawie unikalnych identyfikatorów, które zapewniają złamanie symetrii między nimi. Złożoność algorytmu to maksymalna droga przebyta przez oba roboty do chwili spotkania. Maksimum jest brane po wszystkich parach różnych identyfikatorów przyporządkowanych robotom i wszystkich sposobach przejścia odpowiadających im tras. W pracy sformułowanych zostało szereg własności, które powinny mieć optymalne algorytmy dla rendezvous. Skonstruowałem też ulepszone algorytmy rendezvous na prostej. W tym ośmiokrotnie efektywniejszy algorytm dla znanej odległości między agentami i algorytm o złożoności $O(DL^2)$ dla znanej długości L identyfikatorów. Podałem też, asymptotycznie efektywniejszy niż sformułowany przez poprzednich autorów, algorytm dla identyfikatorów o nieznanych długościach.

Algorytmy online. Algorytmom online poświęcone są cztery prace. W pracach [13, 16, 17] zajmujemy się problemem zbierania przedmiotów o różnych wagach. Przedmioty są ustawione w kolejce i celem jest zebranie przedmiotów o jak największej sumarycznej wadze. W trakcie działania algorytmu przedmioty mogą się pojawiać w dowolnych momentach czasu i znikają w kolejności występowania w kolejce. Algorytm zbierający w każdej rundzie największy istniejący element jest 2-kompetytywny. Dla tego problemu istnieje też proste ograniczenie dolne na kompetytywność $\phi \approx 1.618$, nawet dla przedmiotów o wagach ustawionych w kolejności nierosnącej. Podaliśmy optymalny ϕ-kompetytywny algorytm dla wag nierosnących. Dla wag ustawionych w kolejce dowolnej kolejności podaliśmy algorytm 1.897-kompetytywny. Dla kolejek FIFO podaliśmy algorytm 1.737-kompetytywny. Ponadto pokazaliśmy w ogólnym przypadku nietrywialne ograniczenie dolne 1.637 na kompetytywność algorytmu nawet w przypadku *statycznym*, gdy wszystkie przedmioty pojawiają się w chwili 0. Mój wkład w prace [13, 16] polegał na skonstruowaniu i zanalizowaniu algorytmu ϕ-kompetytywnego w przypadku statycznym, którego

modyfikacjami były wszystkie dalsze algorytmy i udziale w poprawianiu konstrukcji algorytmu FIFO. Mój wkład w pracę [17] polegał na podaniu zupełnie nowej analizy ogólnego algorytmu ϕ-kompetytywnego.

W ogólnym problemie agregacji wiadomości kontrolnych (CMA), pakiety są generowane w pewnych momentach czasu w węzłach drzewa T i muszą być przesłane do korzenia T. Koszt transmisji po krawędzi jest równy jej długości, niezależnie od liczby wysłanych pakietów. Pakiety mogą też czekać na transmisję w węzłach drzewa, co wiąże się z kosztem proporcjonalnym do czasu oczekiwania. W celu zoptymalizowania ogólnych kosztów transmisje te mogą być opóźniane dla skumulowania pakietów i wysłania ich dalej pojedynczą transmisją. Sekwencja transmisji, która przekazuje wszystkie pakiety nazywa się ich *uszeregowaniem* i zawsze istnieje optymalne uszeregowanie złożone z transmisji, które następują od razu po poddrzewach zakorzenionych w korzeniu drzewa. W szczególnym przypadku dla T będącego półprostą (chain network) rozważanym w [20] podaliśmy 5-konkurencyjny algorytm online i dolną granicę $2+\phi \approx 3.618$, poprawiając wyniki poprzednich autorów – odpowiednio: 8 i 2. Ponadto w wersji offline, podaliśmy dokładny algorytm wielomianowy. Mój wkład polegał na udziale w konstrukcji algorytmu online.

Sieci radiowe. Problemy związane z sieciami radiowymi są przedmiotem rozważań w sześciu pracach, których jestem współautorem po doktoracie. Pierwsze z nich [11, 12] dotyczą problemu budzenia w modelu sieci radiowych single hop (to znaczy każdy może się komunikować z każdym) bez wykrywania kolizji. Praktycznym przykładem takiej sieci jest ethernet (który w sensie dosłownym sam nie jest siecią radiową). Węzły sieci mają dostęp do lokalnego zegara, który dostarcza im taktowania potrzebnego do wykonywania obliczeń w rundach ale nie jest (na ogół) zegarem globalnym, który podaje przy każdej rundzie jej numer. Celem protokołów jest uzyskanie rundy w której dokładnie jeden węzeł sieci nadaje. Brak wykrywania kolizji oznacza, że sytuacje kiedy nikt nie nadaje i co najmniej dwie stacje nadają są nierozróżnialne. W pracy rozważamy algorytmy probabilistyczne i cel protokołu jest uzyskiwany z największym prawdopodobieństwem w rundach, w których suma prawdopodobieństw nadawań węzłów jest bliska 1. Interesowały nas algorytmy uzyskujące budzenie z dużym prawdopodobieństwem $1-\varepsilon$ dla dowolnie małego stałego $\varepsilon > 0$ Uzyskaliśmy ograniczenie dolne $O(n/\log n)$ na czas budzenia w najbardziej ogólnym modelu. Podaliśmy też algorytmy budzenia w czasie polilogarytmicznym gdy ogólny model wzbogacimy o jedną z trzech dodatkowych własności: znajomość liczby stacji, unikalne identyfikatory stacji lub globalny zegar. Mój wkład polegał na konstrukcji algorytmów dla unikalnych identyfikatorów oraz globalnego zegara i na podaniu ograniczenia dolnego w najbardziej ogólnym przypadku.

W czterech dalszych pracach konferencyjnych rozważany był model komunikacji radiowej SINR czyli komunikacji o której powodzeniu decyduje współczynnik stosunku SINR (signal to noise and interference ratio) — sygnału do sumy szumu i interferencji. W modelu tym węzły sieci (stacje radiowe) rozłożone są na płaszczyźnie lub w innej przestrzeni metrycznej stanowiącej domyślnie pewne uogólnienie płaszczyzny. Każdy węzeł v może w danej rundzie komunikacji albo nadawać albo nie nadawać. Jeśli węzeł nie nadaje, to może odbierać wiadomości od innych węzłów. Jeśli węzeł v nadaje a u nie nadaje, to stacja u odbiera z v sygnał o mocy $P_v(u) = P_v/d^\alpha(u,v)$, gdzie P_v jest mocą nadajnika v, a $d(u,v)$ jest (euklidesową) odległością między u i v. Stała $\alpha \geq 2$ wyznacza szybkość zanikania sygnału z odległością – stała $\alpha = 2$ odpowiada szybkości zaniku sygnału radiowego w próżni a $\alpha > 2$ odpowiada zmodyfikowanej stałej dla próżni po uwzględnieniu przeszkód w komunikacji.

Niech stacja v nadaje jakąś wiadomość do stacji u. Przeszkodami dla odebrania tej wiadomości są ogólny szum N i sygnały z innych nadających stacji, które w sumie mają moc $I_v(u) = \sum_{w \in T \setminus v} P_w(u)$ gdzie T jest zbiorem nadających stacji. Wiadomość jest odbierana przez u gdy $P_v(u)$ jest przynajmniej β razy większe od sumy $N + I_v(u)$, gdzie β jest stałą większą od

1 wyrażającą czułość odbiornika. Innymi słowy wiadomość jest odbierana dokładnie wtedy, gdy proporcja SINR= $P_v(u)/(N + I_v(u))$ jest większa niż β.

W pracach z modelem SINR przyjmowaliśmy, że rundy komunikacji dokonywane są synchronicznie, stacje mają jednakowe moce nadawania i nie uzyskują w rundzie żadnej innej informacji o komunikacji, niż odebranie lub nieodebranie wiadomości. We wszystkich czterech pracach w modelu SINR rozważaliśmy problem rozgłaszania (broadcasting) w sieciach radiowych. Na początku protokołu dokładnie jeden węzeł sieci jest w posiadaniu pewnej wiadomości m i celem jest to, żeby na końcu wszystkie węzły znały tę wiadomość. Przedmiotem naszego zainteresowania są algorytmy rozproszone. Najbardziej pożądanymi przy tym z naszego punktu widzenia są algorytmy działające w sieciach ad hoc, w których węzły nie wiedzą nic o położeniu innych węzłów w sieci na początku protokołu.

W trzech pracach [19, 18, 21] przyjęliśmy dodatkowo założenie, że węzły znają swoje położenie tzn. współrzędne na płaszczyźnie. Może to być położenie przybliżone obarczone pewnym błędem.

W pracach [19, 18] rozważane są algorytmy deterministyczne. W pracy [19] model komunikacji jest modelem *słabym*, w którym obok warunku SINR wprowadzony jest warunek, że komunikacja na odległość większą niż $1 - \varepsilon$ jest niemożliwa. Pokazaliśmy, że w modelu ad-hoc, w którym węzły nic nie wiedzą o swoim sąsiedztwie jest dolne ograniczenie $\Omega(n)$ na czas rozgłaszania. W przypadku gdy węzły znają wszystkich swoich sąsiadów to jest wszystkie węzły z którymi mogą się komunikować, czas ten można zmniejszyć do $O(D \log^2 n)$, gdzie D jest średnicą sieci. Mój wkład polegał na skonstruowaniu i analizie algorytmu wyboru lidera w oparciu o ogólny schemat zaproponowany przez współautorów.

We wszystkich dalszych pracach rozpatrujemy *silny* model komunikacji, w którym komunikacja możliwa jest zawsze, gdy spełniony jest warunek SINR. Niezależnie od tej własności wymagamy, żeby sieć była spójna jako graf połączeń o długościach nie większych niż $1-\varepsilon$ dla pewnego parametru $\varepsilon > 0$. W pracy [18] podaliśmy deterministyczny algorytm rozgłaszania o złożoności $O(D \log^2 n)$. Mój wkład polegał na skonstruowaniu i analizie algorytmu wyboru lidera. W pracy [21] podaliśmy zrandomizowany algorytm rozgłaszania whp. o złożoności $O(D \log n + \log^2 n)$. Skrót whp. w ostatnim zdaniu oznacza z dużym prawdopodobieństwem (with high probability) czyli z prawdopodobieństwem co najmniej $1 - n^a$ dla z góry ustalonego parametru $a > 0$. Mój wkład polegał na skonstruowaniu i analizie algorytmów rozgłaszania z tej pracy dla znanej gęstości i algorytmu rozgłaszania o złożoności $O(D \log n + \log^2 n)$.

W pracy [15] porzucamy założenie, że węzły znają swoje położenie na płaszczyźnie. Dokonują one decyzji w trakcie algorytmu jedynie na podstawie tego jak często w czasie każdej jego fazy odbierają wiadomości. Podaliśmy w niej zrandomizowany algorytm rozgłaszania whp. o złożoności $O(D \log^2 n)$. Mój wkład polegał na skonstruowaniu i analizie algorytmu rozgłaszania w oparciu o ogólny schemat zaproponowany przez współautorów.

Statystyki:

Znaleziono w WoS (Web of Science): 17 prac z 21
Liczba cytowań wg. WoS: 75
Liczba cytowań bez autocytowań wg. WoS: 68
h-index wg. WoS: 5 (jeśli prace [11, 12] potraktujemy łącznie)
h-index wg. Google Scholar: 8 (łącznie 235 cytowań)
Sumaryczny Impact Factor (JCR): 5,957
W tym w pracach, które ukazały się po doktoracie: 4,393

Literatura

[1] Grzegorz Stachowiak: Fibonacci Correction Networks. SWAT 2000: 535-548

[2] Grzegorz Stachowiak: Fast Periodic Correction Networks. FCT 2003: 144-156

[3] Grzegorz Stachowiak: Lower Bounds on Correction Networks. ISAAC 2003: 221-229

[4] Grzegorz Stachowiak: Fast periodic correction networks. Theor. Comput. Sci. 354(3): 354-366 (2006)

[5] Grzegorz Stachowiak: The Number of Linear Extensions of Bipartite Graphs, Order 5 (1988), 257 - 259.

[6] Grzegorz Stachowiak: A Relation between the Comparability Graph and the Number of Linear Extensions, Order 6 (1989), 241 - 244.

[7] Grzegorz Stachowiak: Hamilton Paths in Graphs of Linear Extensions for Unions of Posets. SIAM J. Discrete Math. 5(2): 199-206 (1992)

[8] Grzegorz Stachowiak: On a long cycle in the graph of all linear extensions of a poset consisting of two disjoint chains. Discrete Mathematics 131(1-3): 375-378 (1994)

[9] Grzegorz Stachowiak: Finding parity difference by involutions. Discrete Mathematics 163(1-3): 139-151 (1997)

[10] Marcin Kik, Miroslaw Kutylowski, Grzegorz Stachowiak: Periodic Constant Depth Sorting Networks. STACS 1994: 201-212

[11] Tomasz Jurdzinski, Grzegorz Stachowiak: Probabilistic Algorithms for the Wakeup Problem in Single-Hop Radio Networks. ISAAC 2002: 535-549

[12] Tomasz Jurdzinski, Grzegorz Stachowiak: Probabilistic Algorithms for the Wake-Up Problem in Single-Hop Radio Networks. Theory Comput. Syst. 38(3): 347-367 (2005)

[13] Marcin Bienkowski, Marek Chrobak, Christoph Durr, Mathilde Hurand, Artur Jez, Lukasz Jez, Grzegorz Stachowiak: Collecting Weighted Items from a Dynamic Queue, SODA 2009, 1126-1135.

[14] Grzegorz Stachowiak: Asynchronous Deterministic Rendezvous on the Line, SOFSEM 2009, 497-508.

[15] Tomasz Jurdzinski, Dariusz R. Kowalski, Michal Rozanski, Grzegorz Stachowiak: On the impact of geometry on ad hoc communication in wireless networks. PODC 2014: 357-366

[16] Marcin Bienkowski, Marek Chrobak, Christoph Dürr, Mathilde Hurand, Artur Jez, Lukasz Jez, Grzegorz Stachowiak: Collecting Weighted Items from a Dynamic Queue. Algorithmica 65(1): 60-94 (2013)

[17] Marcin Bienkowski, Marek Chrobak, Christoph Dürr, Mathilde Hurand, Artur Jez, Lukasz Jez, Grzegorz Stachowiak: A ϕ-competitive algorithm for collecting items with increasing weights from a dynamic queue. Theor. Comput. Sci. 475: 92-102 (2013)

[18] Tomasz Jurdzinski, Dariusz R. Kowalski, Grzegorz Stachowiak: Distributed Deterministic Broadcasting in Uniform-Power Ad Hoc Wireless Networks. FCT 2013: 195-209

[19] Tomasz Jurdzinski, Dariusz R. Kowalski, Grzegorz Stachowiak: Distributed Deterministic Broadcasting in Wireless Networks of Weak Devices. ICALP 2013: 632-644

[20] Marcin Bienkowski, Jaroslaw Byrka, Marek Chrobak, Lukasz Jez, Jiri Sgall, Grzegorz Stachowiak: Online Control Message Aggregation in Chain Networks. WADS 2013: 133-145

[21] Tomasz Jurdzinski, Dariusz R. Kowalski, Michal Rozanski, Grzegorz Stachowiak: Distributed Randomized Broadcasting in Wireless Networks under the SINR Model. DISC 2013: 373-387

[22] Miklós Ajtai, János Komlós, Endre Szemerédi: Sorting in c log n parallel sets. Combinatorica 3(1): 1-19 (1983)

[23] Kenneth E. Batcher: Sorting Networks and Their Applications. AFIPS Spring Joint Computing Conference 1968: 307-314

[24] T.H.Cormen, C.E.Leiserson, R.L.Rivest: Wprowadzenie do algorytmów, WNT (2005).

[25] Marcin Kik, Miroslaw Kutylowski, Marek Piotrów: Correction Networks. ICPP 1999: 40-47

[26] Marcin Kik: Periodic Correction Networks. Euro-Par 2000: 471-478

[27] Donald Knuth: Sztuka programowania Tom IV, Zeszyt 2: Generowanie wszystkich krotek i permutacji. WNT (2007).

[28] Miroslaw Kutylowski, Krzysztof Lorys, Brigitte Oesterdiekhoff, Rolf Wanka: Fast and Feasible Periodic Sorting Networks of Constant Depth FOCS 1994: 369-380

[29] Marek Piotrów: Depth Optimal Sorting Networks Resistant to k Passive Faults. SIAM J. Comput. 33(6): 1484-1512 (2004)

[30] Marek Piotrów: Periodic, random-fault-tolerant correction networks. SPAA 2001: 298-305

[31] Manfred Schimmler, Christoph Starke: A Correction Network for N-Sorters. SIAM J. Comput. 18(6): 1179-1187 (1989)

[32] Andrew Chi-Chih Yao, F. Frances Yao: On Fault-Tolerant Networks for Sorting. SIAM J. Comput. 14(1): 120-128 (1985)