Załącznik 2a: Autoreferat

[1] Imię i nazwisko

Aleksandra Świercz

[2] Posiadane stopnie i tytuły naukowe

1. **Tytuł inżyniera informatyki** uzyskany w 2000 roku na Wydziale Elektrycznym Politechniki Poznańskiej (tytuł pracy dyplomowej inżynierskiej: „Szeregowanie zadań w systemach obsługi w środowisku programistycznym CHIP"; promotor: prof. dr hab. inż. Jacek Błażewicz)
2. **Tytuł magistra informatyki** uzyskany w 2002 roku na Wydziale Elektrycznym Politechniki Poznańskiej (tytuł pracy dyplomowej magisterskiej: „Algorytmy przybliżone sekwencjonowania łańcuchów DNA z wykorzystaniem bibliotek izotermicznych", promotor prof. dr hab. inż. Marta Kasprzak)
3. **Stopień doktora nauk technicznych** w zakresie informatyki, uzyskany w 2007 roku na Wydziale Informatyki i Zarządzania Politechniki Poznańskiej (tytuł rozprawy doktorskiej: „Sekwencjonowanie genomu: algorytmy dla nowych podejść"; promotor: prof. dr hab. inż. Marta Kasprzak)

[3] Dotychczasowe zatrudnienie w jednostkach naukowych

- 2000 – 2005: asystent techniczny, Instytut Informatyki, Politechnika Poznańska
- 2005 – 2008: asystent naukowy, Instytut Informatyki, Politechnika Poznańska
- 2005 – 2007: asystent naukowy, Instytut Chemii Bioorganicznej Polskiej Akademii Nauk w Poznaniu
- 2007 – 2015: adiunkt, Instytut Chemii Bioorganicznej Polskiej Akademii Nauk w Poznaniu
- od 2008: adiunkt, Instytut Informatyki, Politechnika Poznańska
- Od 2015: starszy specjalista, Instytut Chemii Bioorganicznej PAN w Poznaniu
- 11-12.2003, 07-08.2004: asystent naukowy, Hong Kong Polytechnic University, Hong Kong
- 10.2010, 10.2011, 11.2012, 04.2016, 10.2017, 10.2018: naukowiec wizytujący, Instituto di Analisi dei Sistemi ed Informatica „Antonio Ruberti", Consiglio Nazionale delle Ricerche, Rzym, Włochy

[4] Wskazanie osiągnięcia naukowego wynikającego z ustawy o stopniach naukowych i tytule naukowym oraz o stopniach i tytule w zakresie sztuki

4.1. Tytuł osiągnięcia naukowego

Metody meta- i hiper-heurystyczne wykorzystywane w procesie odczytywania genomu.

4.2. Lista prac wchodzących w skład osiągnięcia naukowego

[P1] A. Swiercz, chapter 1, "Hyper-Heuristics and Metaheuristics for Selected Bio-Inspired Combinatorial Optimization Problems" in J. Del Ser Lorente (ed.) "Heuristics and Hyper-Heuristics - Principles and Applications", InTech, pp.3-20, 2017

[P2] A. Swiercz, W. Frohmberg, M. Kierzynka, P. Wojciechowski, P. Zurkowski, J. Badura, A. Laskowski, M. Kasprzak, J. Blazewicz, „GRASShopPER – an algorithm for de novo assembly based on GPU alignments", PLOS ONE, 13(8): e0202355. 2018

[P3] J. Blazewicz, M. Kasprzak, M. Kierzynka, W. Frohmberg, A. Swiercz, P. Wojciechowski, P. Zurkowski, "Graph algorithms for DNA sequencing - origins, current models and the future", European Journal of Operational Research 264 (3), 2018, pp.799-812.

[P4] A. Swiercz, E.K. Burke, M. Cicheński, G. Pawlak, S. Petrovic, T. Zurkowski, J. Blazewicz, "Unified encoding for hyper-heuristics with application to bioinformatics" Central European Journal of Operations Research 22, 2014, pp.567-589.

[P5] A. Swiercz, B. Bosak, M. Chlopkowski, A. Hoffa, M. Kasprzak, K. Kurowski, T. Piontek, J. Blazewicz, "Preprocessing and storing high-throughput sequencing data", Computational Methods in Science and Technology 20, 2014, pp.9-20.

[P6] J. Blazewicz, W. Frohmberg, P. Gawron, M. Kasprzak, M. Kierzynka, A. Swiercz, P. Wojciechowski, "DNA sequence assembly involving an acyclic graph model", Foundations of Computing and Decision Sciences 38, 2013, pp.25-34.

[P7] J. Blazewicz, E.K. Burke, G. Kendall, W. Mruczkiewicz, C. Oguz, A. Swiercz, "A hyper-heuristic approach to sequencing by hybridization of DNA sequences", Annals of Operations Research 207(1), 2013, pp. 27-41.

[P8] J. Blazewicz, M. Bryja, M. Figlerowicz, P. Gawron, M. Kasprzak, E. Kirton, D. Platt, J. Przybytek, A. Swiercz, L. Szajkowski, „Whole genome assembly from 454 sequencing output via modified DNA graph concept", Computational Biology and Chemistry 33, 2009, 224–230.

[P9] M. Kasprzak, A. Swiercz, „Sekwencjonowanie i asemblacja DNA - podejścia, modele grafowe, algorytmy" Kosmos, 58 no. 1-2, 2009, pp. 17-28.

4.3. Omówienie celu osiągnięcia naukowego w/w prac oraz osiągniętych wyników wraz z omówieniem ich ewentualnego wykorzystania

4.3.1. Wprowadzenie

Od wielu lat ludzie próbowali rozwikłać zagadkę, jaki czynnik jest odpowiedzialny za to w jaki sposób funkcjonują organizmy, za to że ludzie różnią się od siebie oraz od innych żywych organizmów. Odkrycie kodu genetycznego zapisanego pod postacią dwuniciowej helisy DNA (kwasu deoksyrybonukleinowego) w 1953 roku przez Jamesa Watsona i Francisa Cricka [WC53] przybliżyło rozwikłanie tejże zagadki i doprowadziło do rozwoju metod rozszyfrowywania kodu. Niestety obecnie nie istnieje żadna technologia, dzięki której można by przeczytać od razu cały genom, czyli poznać kolejność nukleotydów w helisie DNA reprezentowanych w sekwencji jako litery: A,C,G,T. Możliwe jest jedynie odczytywanie krótszych fragmentów, złożonych z kilkuset do kilku tysięcy liter, a następnie złożenie ich w długą sekwencję genomu. Ze względu na niesłychaną złożoność oraz długość genomów organizmów żywych okazało się szybko, że ręczne składanie takich fragmentów jest czasowo nieosiągalne, udowodniono że problem ten (a właściwie jego podproblem znalezienia najkrótszej wspólnej nadsekwencji danego zbioru sekwencji) jest NP-trudny [GMA80]. Dla przykładu genom człowieka jest zbudowany z około 3 miliardów nukleotydów, podzielonych na 23 fragmenty zwane chromosomami. Odczytywanie genomu zapoczątkowało rozwój metod matematycznych oraz informatycznych wspomagających proces przetwarzania wielomilionowych zbiorów krótkich sekwencji.

Odczytywanie genomu można podzielić na kilka etapów: sekwencjonowanie, asemblację oraz wykańczanie genomu. *Sekwencjonowanie* polega na bezpośrednim czytaniu fragmentów sekwencji. Dwie pierwsze chronologicznie metody polegały na wykorzystaniu metod żelowych, znane jako

metoda Sangera [SNC77] oraz metoda Maxama-Gilberta [MG77]. Kolejną była metoda Southerna [Sou88], znana pod nazwą sekwencjonowanie przez hybrydyzację (SBH). Wszystkie wymienione metody odczytywały jednorazowo sekwencję o długości kilkuset nukleotydów, choć trzecia z nich wymagała jeszcze dodatkowo *fazy obliczeniowej* w celu połączenia kilkunukleotydowych fragmentów sekwencji w dłuższy łańcuch. Metody, choć bardzo kosztowne i czasochłonne, przyczyniły się między innymi do poznania genomu ludzkiego w projekcie *Human Genome Project* [IHGSC01,HGP03]. Z początkiem XXI wieku nastąpił rozwój metod sekwencjonowania tzw. nowej generacji, które znacząco przyspieszają odczyt, zmniejszają koszt oraz pozwalają na odczytanie jednocześnie wiele milionów sekwencji o długości do kilkuset nukleotydów. W chwili obecnej obserwujemy rozwój metod trzeciej generacji polegających na odczytywaniu jednorazowo sekwencji o długości do kilkunastu tysięcy nukleotydów, lecz zawierające znacznie więcej błędów niż w przypadku poprzednich metod.

W drugim etapie odczytywania genomów - *asemblacji*, łączone są sekwencje otrzymane w trakcie sekwencjonowania. Informatyczne metody asemblacji, tzw. asemblery, powinny w efektywny sposób przetworzyć miliony sekwencji, zbadać ich podobieństwo i stopień nałożenia oraz utworzyć długi ciąg znaków, zwany *kontigiem*, zawierający jako podsekwencje wszystkie sekwencje wejściowe. Ze względu na trudność problemu, asemblery nie są w stanie odtworzyć jednej długiej, oryginalnej sekwencji genomu, co w powoduje powstanie wielu kontigów wynikowych. Stąd wynika konieczność kolejnego etapu – *wykańczania*, w którym to porządkowane są kontigi, ustalana jest odległość między nimi i pozycjonowanie na odpowiednim chromosomie. Wykorzystywane są w tym celu różne metody, najczęściej biochemiczne, takie jak na przykład znakowanie czy mapy optyczne.

W ramach prezentowanych w cyklu habilitacyjnym wyników zamodelowałam metody hiper-heurystyczne dla problemu sekwencjonowania przez hybrydyzację; była to kontynuacja badań podjętych na etapie doktoratu [A6, A7, A8]. Rozwiązanie problemów sekwencjonowania w małej skali, umożliwiło następnie rozwiązanie problemu sekwencjonowania w dużej skali, z wielokrotnie większą liczbą dłuższych niż w przypadku SBH sekwencji, które należy ze sobą połączyć, czyli problemu asemblacji DNA. Prowadziłam również badania nad specyfiką grafów wykorzystywanych do zamodelowania obu problemów. Byłam kierownikiem grantów, dotyczących sekwencjonowania w dużej skali, oraz integracji danych z sekwencjonowania z informacją zawartą w innych bazach danych (ekspresja genów, ontologia genów, interakcji białko-białko). Ponadto brałam udział w wielu projektach, w których jako ekspert bioinformatyczny przeprowadzałam analizę danych z sekwenatorów, co zaowocowało kilkoma publikacjami [A1, A2, A3, A4] oraz zgłoszeniem patentu [C1]. Zdobyta w trakcie prowadzonych badań wiedza, dotycząca specyfiki danych z sekwencjonowania oraz rodzaju błędów w sekwencjach, została wykorzystana podczas konstrukcji algorytmów asemblacji *de novo*.

Omówienie głównych celów i wyników naukowych wszystkich prac składających się na prezentowany cykl zostanie podzielony na cztery komplementarne części, grupujące uzyskane wyniki:

i. Problem sekwencjonowania przez hybrydyzację. Przedstawienie różnych modeli grafowych wykorzystywanych w kontekście odczytywania genomu oraz zależności pomiędzy nimi [P3,P9].

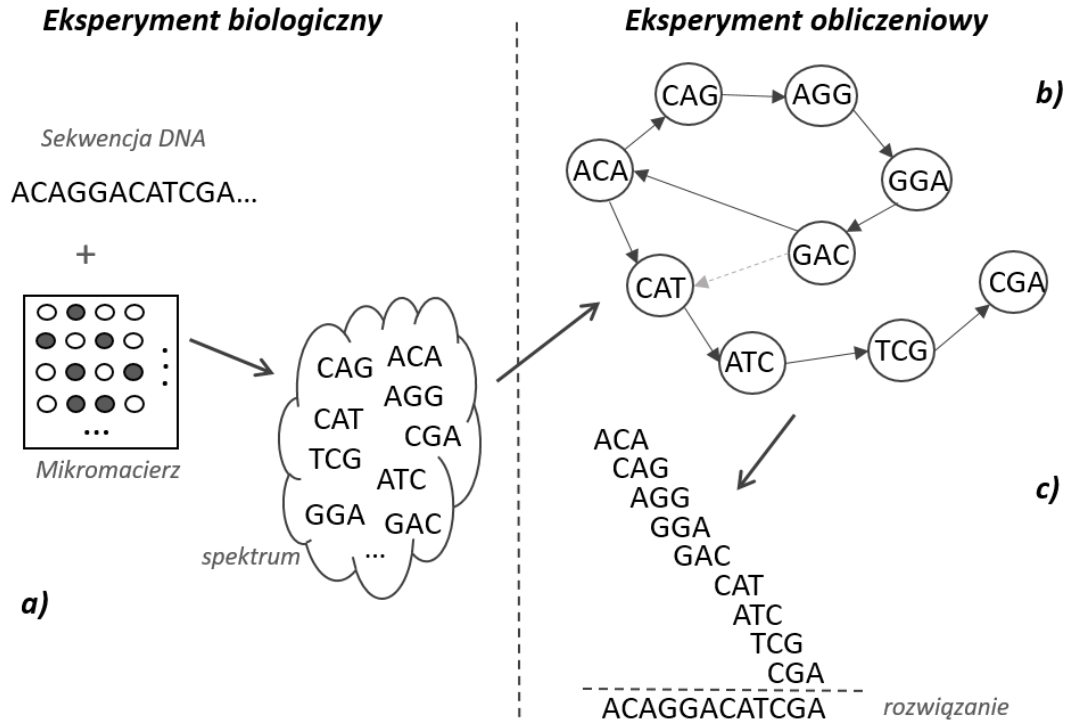
ii. Przedstawienie klasyfikacji hiper-heurystyk ze względu na sposób konstrukcji rozwiązania oraz sposób uczenia [P1], opracowanie modelu hiper-heurystyki dla problemu sekwencjonowania przez hybrydyzację [P7] oraz modelu jednorodnego kodowania dla różnych problemów kombinatorycznych i przeniesienia bariery domenowej w hiper-heurystyce [P4].

iii. Opracowanie różnych algorytmów do asemblacji DNA: z wykorzystaniem obliczeń na kartach graficznych [P2], z grafem acyklicznym [P6], oraz przy użyciu modelu zmodyfikowanego grafu DNA [P8]. Zastosowanie hiper-heurystyki w problemie asemblacji DNA [P1,P2].

iv. Kompresja danych z sekwencjonowania w celu usprawnienia działania algorytmów asemblacji oraz transferu danych na serwer [P5].

4.3.2. Sekwencjonowanie przez hybrydyzację

Sekwencjonowanie przez hybrydyzację jest metodą odczytywania krótkich fragmentów DNA, która się składa z eksperymentu biochemicznego oraz fazy obliczeniowej. W pierwszej części tworzona jest mikromacierz, na której umieszczone są wszystkie sekwencje o zadanej długości *k* (*k*-mery), np. *k*=8. Z drugiej strony przygotowywana jest, będąca celem odczytu, kilkusetnukleotydowa sekwencja DNA, namnożona w wielu milionach kopii, oznakowana fluorescencyjnie. W trakcie eksperymentu biochemicznego sekwencja DNA przyczepia się (hybrydyzuje) do komplementarnych *k*-merów na mikromacierzy, a skanowanie obrazu mikromacierzy pozwala na detekcję świecących fluorescencyjnie punktów, w których nastąpiła hybrydyzacja. Znając pozycję poszczególnych *k*-merów jesteśmy w stanie odtworzyć zbiór wszystkich *k*-merów, które są podsekwencjami badanej sekwencji DNA. Zbiór ten nazywa się spektrum (por. Rys. 1). W kolejnym etapie należy odtworzyć sekwencję DNA z *k*-merów sprawdzając ich nakładanie się. Większość zaproponowanych w literaturze metod korzysta z dwóch rodzajów grafów. W zaproponowanym przez Lysova [LFK+88] grafie *G*, każdy z *k*-merów jest wierzchołkiem, a łuki łączą wierzchołki, których etykiety nakładają się na siebie z przesunięciem o 1. Rozwiązaniem jest ścieżka Hamiltona. W roku 1989 Pevzner [Pev89] zaproponował graf *H*, w którym *k*-mery są łukami łączącymi wierzchołki o długości *k*-1, które są sufiksem i prefiksem etykiety łuku. Rozwiązaniem problemu SBH w grafie *H* jest ścieżka Eulera. Związek między grafami *H* oraz *G* został wyjaśniony w 1999 roku [BHK+99], a podsumowanie zagadnienia dotyczącego tychże grafów i ich wykorzystania w kontekście poznawania genomu zostało zawarte w pracy [P3].



Rys. 1. SBH składa się z biologicznego i obliczeniowego eksperymentu. W pierwszym (a) na podstawie mikromacierzy znajdowane są wszystkie *k*-mery będące podsekwencjami sekwencji DNA. W drugiej części każdy element spektrum staje się wierzchołkiem w grafie (b) Łuki z pełną kreską wskazują na nakładanie się połączonych wierzchołków z przesunięciem o 1, natomiast zakropkowane łuki z przesunięciem o 2 (większość z tych łuków została pominięta, aby zachować czytelność rysunku). Po znalezieniu ścieżki Hamiltona w grafie jesteśmy w stanie odtworzyć sekwencję DNA. Z uwagi na błąd negatywny ACA można uzyskać również krótsze rozwiązanie utworzone z takiej samej liczby *k*-merów rozpoczynające się od CAGG…

Grafy Lysova *G*, nazywane także grafami DNA, należą do grafów skierowanych etykietowanych. Są one także liniowymi grafami skierowanymi, dla których istnieje wielomianowe rozwiązanie poszukiwania

ścieżki Hamiltona. Jest to możliwe dzięki transformacji liniowego grafu skierowanego do jego grafu oryginalnego (graf Pevznera) i szukaniu ścieżki Eulera w tym drugim grafie. Liniowy graf skierowany *G* oraz jego graf oryginalny *H* są połączone ze sobą w następujący sposób:

- wierzchołki w grafie *G* odpowiadają łukom w grafie *H*
- istnieje łuk (*i*,*j*) w *G* wtedy i tylko wtedy, gdy w grafie *H* końcowy wierzchołek łuku *i* jest również wierzchołkiem początkowym łuku *j*

Istnienie ścieżki Eulera w *H* jest równoważne istnieniu ścieżki Hamiltona w *G*.

W kontekście poznawania genomu często używa się, niesłusznie, nazwy grafy de Bruijna. Grafy te są również etykietowane, utworzone ze wszystkich możliwych etykiet o zadanej długości *k*, nad pewnym alfabetem, w przypadku DNA – cztero-literowym {A,C,G,T}. Zatem w grafie mamy 4^k wierzchołków z różnymi etykietami. Łuk łączy wierzchołki, jeśli sufiks poprzednika o długości *k*-1 pokrywa się z prefiksem następnika [B46]. Grafy DNA są indukowanymi wierzchołkowo (vertex-induced) podgrafami grafów de Bruijna. Natomiast grafy Pevznera są podgrafami grafów DNA, są zatem niejako podgrafami grafów de Bruijna. Nie są natomiast grafami de Bruijna, gdyż nie zawierają wszystkich łuków, w których etykiety nakładają się ze sobą. Podgrafy grafów de Bruijna (w części 4.3.4 nazywane grafami dekompozycji) wykorzystywane są jako jedno z podejść do asemblacji genomów.

Grafy Lysova oraz grafy Pevznera zaproponowane zostały dla rozwiązania problemu SBH w przypadku idealnym, który jak powyżej wyjaśniono jest problemem rozwiązywalnym w czasie wielomianowym. Jednakże w przypadku błędów w spektrum, zarówno błędów pozytywnych (dodatkowe *k*-mery w spektrum, które nie są podsekwencją badanej sekwencji DNA), jak i błędów negatywnych (brak *k*-merów w spektrum) problem jest silnie NP-trudny [BK03]. W prowadzonych badaniach przyjęto następującą definicję problemu SBH [P7]:

> <u>*Parametry*</u>: Spektrum *S*, czyli zbiór fragmentów DNA o równej długości *k* (*k*-mery) oraz długość *n* badanej sekwencji
>
> <u>*Odpowiedź*</u>: Sekwencja o długości ≤ *n* składająca się z maksymalnej liczby elementów z *S.*

W podanej definicji dopuszcza się oba typy błędów w spektrum, stąd też w rozwiązaniu nie trzeba wykorzystać wszystkich k-merów (ze względu na błędy pozytywne) oraz przesunięcie między sąsiadującymi k-merami w rozwiązaniu może być większe niż 1 (ze względu na błędy negatywne). Warto zauważyć tutaj iż powtórzenie *k*-meru w sekwencji spowoduje błąd negatywny, gdyż każdy *k*-mer może pojawić się w spektrum tylko jeden raz. Problem sekwencjonowania przez hybrydyzację z błędami został sformułowany jako komiwojażera z nagrodami [BFK+99a], w którym wierzchołki w pełnym grafie są zaetykietowane *k*-merami, a do łuków przypisany jest koszt równy przesunięciu pomiędzy etykietami sąsiadujących wierzchołków. Odwiedzenie każdego z wierzchołków zwiększa zysk na trasie o 1. W takim grafie poszukiwana jest ścieżka prosta o największym zysku, a koszcie nie większym niż *n* – *k.*

W pracy [P9] przedstawiono i porównano różne podejścia i modele grafowe dla problemu SBH. W klasycznym ujęciu do SBH używa się fragmentów o równej długości *k*. Ze względu na sporą liczbę błędów w trakcie eksperymentu biochemicznego zaproponowane zostały inne podejścia do SBH. Preparata i in. [PFU99, PU01] zaproponowali wykorzystanie dłuższych *k*-merów, ale ze względu na to, iż liczba różnych *k*-merów, które musiały by się zmieścić na mikromacierzy, wzrasta wykładniczo wraz ze wzrostem *k*, nukleotydy przeplatane byłyby na zmianę z zasadami uniwersalnymi, co pozwoliłoby wydłużyć sekwencje bez wzrostu ich liczności. Zasady uniwersalne są to takie cząsteczki chemiczne, które mogą się przykleić do dowolnego ze standardowych nukleotydów stojącego po drugiej stronie helisy DNA. Jeden ze specjalnych wzorców sekwencji (ang. g*apped probe*) wygląda w następujący sposób $GP(s,r) = X^s(U^{s-1}X)^r$, gdzie X jest normalną zasadą, a U uniwersalną. Dla przykładu wzorzec GP(3,2) wygląda tak: XXXUUXUUX, a sekwencja o takim wzorcu, np. GAAUUTUUG. W ten sposób można uzyskać znacznie dłuższe fragmenty DNA niż o długości *k*, przy stałym rozmiarze mikromacierzy. Pozwala to uniknąć błędów negatywnych pochodzących z powtórzeń krótkich fragmentów DNA. Inne podejście z wykorzystaniem bibliotek izotermicznych fragmentów DNA ma na celu zmniejszenie

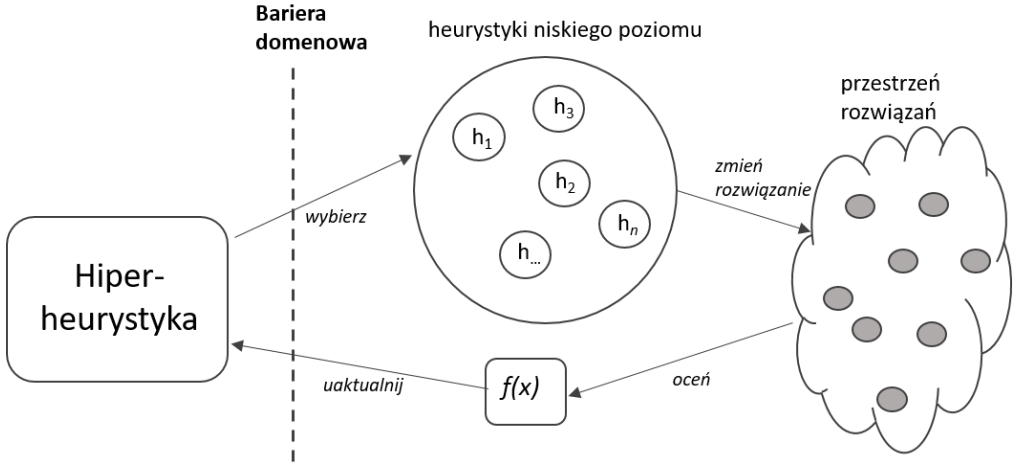
błędów z eksperymentu biochemicznego poprzez wyrównanie temperatury hybrydyzacji fragmentów. Zauważono, że nie wszystkie wiązania w helisie DNA są równie silne, przyjęto założenie, że siła wiązania G/C dwukrotnie przewyższa wiązanie A/T [WJH+81]. Ten model pozwolił zrekompensować niższą stabilność dwuniciowych fragmentów DNA bogatych w pary A/T. Aby wyznaczyć temperaturę fragmentu DNA przyjmuje się że każdy nukleotyd G lub C zwiększa jego temperaturę o 4 stopnie, a A oraz T o 2 stopnie. Dla przykładu fragmentem DNA o temperaturze 12 stopni mogą być następujące sekwencje: ATATATAT, GGC oraz GATC. Pokazano, iż aby pokryć wszystkie sekwencje DNA podczas eksperymentu hybrydyzacji potrzebne są dwie biblioteki fragmentów z temperaturami różniącymi się o 2 stopnie [BFK+99b]. Dla podejścia izotermicznego zaproponowane zostały metody heurystyczne [A6, A7, A8], składające się na moją pracę doktorską.

Zgłębienie tematyki różnych podejść, modeli grafowych i algorytmów dla problemu SBH zaowocowało w późniejszym czasie rozwijaniem nowych podejść w kolejnym etapie poznawania genomu, czyli asemblacji (por. sekcja 4.3.4).

Badania prowadzone przed uzyskaniem stopnia doktora skupiły się na konstrukcji metod metaheurystycznych dla klasycznego i izotermicznego problemu SBH [A6,A7,A8]. W badaniach przedstawionych w sekcji 4.3.3 skupiono się na metodach hiper-heurystycznych, oraz porównaniu z najlepszymi metodami rozwiązującymi problem SBH.

4.3.3. Zastosowanie hiper-heurystyk w problemie sekwencjonowania przez hybrydyzację

Termin hiper-heurystyka pierwszy raz został użyty przez Cowlinga i in. w 2001 [CKS01] w odniesieniu do metody, która nie wykorzystuje wiedzy dotyczącej rozwiązywanego problemu, lecz korzysta z prostych, łatwych do zaimplementowania heurystyk. Pomiędzy zbiorem prostych heurystyk niskiego poziomu a heurystyką wysokiego poziomu istnieje bariera domenowa, która nie pozwala na przesyłanie informacji o domenie problemu. Schemat działania hiper-heurystyki przedstawia Rys. 2. Hiper-heurystyka na podstawie wskaźników jak zachowywały się heurystyki niskiego poziomu powinna zdecydować która heurystyka powinna zostać wywołana w danym punkcie czasowym, aby znaleźć jak najlepsze rozwiązanie. Celem działania hiper-heurystyki jest znalezienie dobrego rozwiązania niskim nakładem kosztów (tworzenie mało skomplikowanych heurystyk) oraz czasu (obliczenia można przerwać w razie konieczności i zwracane jest najlepsze rozwiązanie spośród znalezionych do tej pory). Dla niektórych problemów kluczowe jest jednak znalezienie optymalnego rozwiązania. Wówczas konstruowane są metaheurystyki „szyte na miarę" dla konkretnego problemu, które pozwalają znaleźć często lepsze rozwiązanie jednakże odbywa się to zdecydowanie większym kosztem.

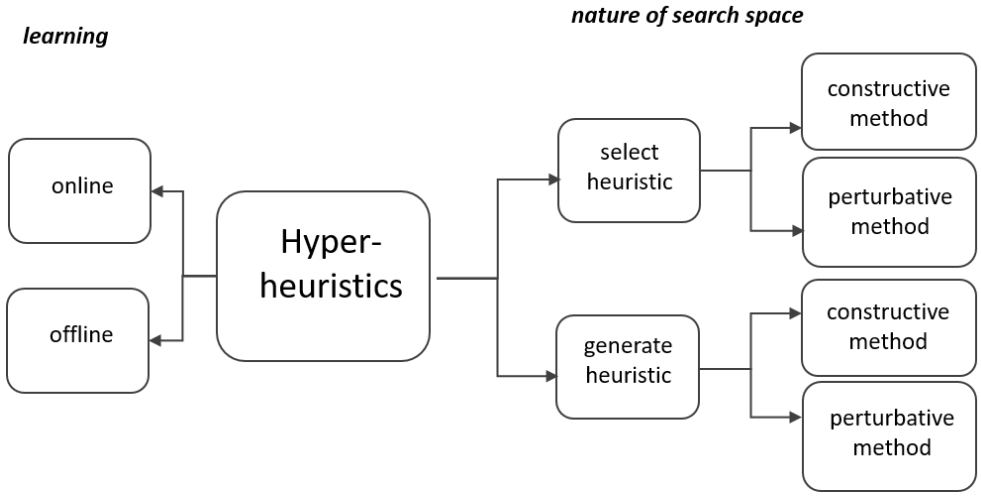


Rys. 2. Sposób działania hiper-heurystyki

Przegląd metod hiper-heurystycznych oraz ich sposób działania opublikowany został na przykład w [BGH+13, POH+14] oraz w [P1]. W pracy [P1] hiper-heurystyki przedstawione były w kontekście przenikania wiedzy między różnymi dziedzinami nauk: informatyki, biologii oraz badań operacyjnych.

Ogólnie przyjęty podział metod hiper-heurystycznych przedstawia schemat na Rys.3. Z jednej strony mamy sposób przeszukiwania przestrzeni, z drugiej sposób uczenia się metody. Jeśli chodzi o sposób przeszukiwania przestrzeni, a zwrócić tu trzeba w szczególności uwagę na to iż nie przeszukujemy przestrzeni rozwiązań lecz przestrzeń heurystyk niskiego poziomu (które z kolei przeszukują przestrzeń rozwiązań), to dzielimy metody na takie, które:

- *wybierają* jedną z istniejących już heurystyk
- *generują* heurystyki z mniejszych, istniejących już komponentów.



Rys. 3. Podział hiper-heurystyk ze względu na sposób uczenia się oraz sposób przeszukiwania przestrzeni heurystyk niskiego *poziomu oraz rozwiązań* (na podstawie [P1])

Drugi podział jest ze względu na sposób tworzenia rozwiązania przez heurystyki niskiego poziomu:

- *konstruujące* – początkowe częściowo utworzone rozwiązanie jest rozbudowywane w kolejnych iteracjach o nowe elementy do momentu uzyskania końcowego rozwiązania
- *zaburzające* (perturbative) - pełne rozwiązanie początkowe jest modyfikowane w kolejnych iteracjach i poszczególne komponenty rozwiązania mogą być przestawione, usunięte itp.

Ostatni podział hiper-heurystyk dotyczy sposobu uczenia się:

- *online* – zdobywanie wiedzy odbywa się w trakcie rozwiązywania konkretnej instancji problemu
- *offline* – wiedza w postaci reguł albo programów zgromadzona jest na początku na podstawie wstępnie przeprowadzonych testów a następnie wykorzystana jest do rozwiązania nowych instancji.

Kolejnym ważnym elementem jest sposób akceptacji rozwiązania. Czy każde rozwiązanie jest akceptowalne, czy też prawdopodobieństwo akceptacji zależne jest np. od liczby iteracji? Hiper-heurystyki z bardzo dobrym rezultatem wykorzystuje się do rozwiązania różnych problemów kombinatorycznych: szeregowania [CRB10, Pil09], ustawiania harmonogramów pracowniczych [BMM+07, SAQ+12], routingu [PR07] czy też w problemach komiwojażera [KP07, Run09]. Jednakże niektóre metody nie wpasowują się w ogólnie przyjęty schemat i w swoim działaniu obejmują kilka kategorii, na przykład wykorzystują jednocześnie metody konstruujące i zaburzające rozwiązanie [GR10], albo jednocześnie metody wybierające i generujące heurystyki [KG04].

Ponadto w pracy [P1] przedstawiłam hiper-heurystyki, które z powodzeniem zostały użyte do rozwiązania problemów biologicznych: sekwencjonowania przez hybrydyzację [P4, P7] oraz w problemie znajdowania najdłuższej wspólnej podsekwencji [TM12], który wykorzystywany jest np. przy porównywaniu sekwencji DNA lub RNA. Dodatkowo przedstawiłam również koncepcję zastosowania

hiper-heurystyki w problemie sekwencjonowania w dużej skali, czyli asemblacji *de novo*, co zostanie szerzej omówione w sekcji 4.3.4.

W pracy [P7] przeprowadziłam wraz z zespołem obszerne badania nad wpływem mechanizmu uczącego w algorytmie hiper-heurystycznym, a zbieżnością do rozwiązania optymalnego. Ogólny schemat każdej hiper-heurystyki zgodny jest z Rys.2 i może zostać zapisany pod postacią pseudokodu:

HYPERHEURISTICSEARCH*(H,ω_0)*
1 t ← 0
2 **while** termination conditions are not satisfied
3 **do** *h* ← SELECTHEURISTIC(*H*)
4 **if** ACCEPTHEURISTIC(*h*) = TRUE
5 **then** $\omega_t \leftarrow h(\omega_{t-1})$
6 **else** $\omega_t \leftarrow \omega_{t-1}$
7 $t \leftarrow t+1$

W omawianej pracy w algorytmie HYPERHEURISTICSEARCH zbiór heurystyk niskiego poziomu *H* oraz rozwiązanie początkowe ω_0 są danymi wejściowymi do programu. Rozwiązanie początkowe tworzone jest z losowo ułożonej listy *k*-merów ze spektrum nie przekraczającej długości *n* oraz zbioru niewykorzystanych *k*-merów. W każdej iteracji *t* wybierana jest jedna heurystyka h_i i jeśli zostanie zaakceptowana, to przyjmowane jest nowe rozwiązanie ω_t, $\omega_t = h(\omega_{t-1})$. W przeciwnym wypadku, rozwiązanie pozostaje niezmienione. Funkcją celu jest maksymalizacja liczby *k*-merów w rozwiązaniu, więc funkcja, która ocenia nowe rozwiązanie, sprawdza liczbę elementów w rozwiązaniu i porównuje z poprzednią.

W fazie wyboru SELECTHEURISTIC zaimplementowane zostały trzy algorytmy: funkcja wyboru (*choice function*), przeszukiwanie tabu oraz symulowane wyżarzanie. Funkcja wyboru [CKS01] wykorzystuje trzy funkcje pomocnicze, które oceniają heurystyki niskiego poziomu:

$$F(t,h) = \alpha f_1(t,h) + \beta f_2(t,h) + \gamma f_3(t,h)$$

Dla każdej heurystyki *h* wyznaczana jest informacja dotycząca jej efektywności w ostatnim czasie (f_1), informacja dotycząca jej ostatniej efektywności w kontekście użycia pary heurystyk, jedna po drugiej (f_2), oraz informacja o tym kiedy ostatnio heurystyka była wybrana (f_3). Funkcje f_1 oraz f_2 mają za zadanie zintensyfikować wybór najlepszej heurystyki, natomiast f_3 wprowadzić urozmaicenie poprzez użycie heurystyki dawno nie wybieranej. Funkcja wyboru tworzy ranking wszystkich heurystyk niskiego poziomu i na podstawie jednej z czterech strategii wybiera jedną z nich:

- STRAIGHTC – algorytm wybiera heurystykę z najlepszą wartością *F*
- RANKEDC – wyznaczane jest *m* najlepszych w rankingu heurystyk i wybierana jest heurystyka z najlepszym wynikiem funkcji celu
- DECOMPC – wybierana jest heurystyka z najlepszym wynikiem funkcji celu spośród tych z najlepszą wartością F, f_1, f_2 lub f_3
- ROULETTEC – wybierana jest heurystyka z prawdopodobieństwem proporcjonalnym do wartości *F*

Kolejną zamodelowaną w pracy [P7] hiper-heurystyką było przeszukiwanie tabu TABUS. Każda heurystyka początkowo ma przypisaną wagę równą 0. W czasie każdej iteracji heurystyka z najwyższą wagą jest wybierana i jeśli poprawia funkcję celu to jej waga jest zwiększana o 1. Jeśli natomiast funkcja celu pogorszy się to waga jest zmniejszana, a heurystyka wrzucana jest na listę tabu i nie może zostać wybrana przez *m* iteracji. Dla wymienionych wyżej hiper-heurystyk wybrana heurystyka jest zawsze akceptowana (w linii 4 pseudokodu).

W podobny sposób heurystykom niskiego poziomu przypisywane są wagi w ostatniej zaprojektowanej hiper-heurystyce – symulowanym wyżarzaniu SIMANN. Algorytm wybiera heurystykę z ważonym prawdopodobieństwem, proporcjonalnym do przypisanej jej wagi. Mechanizm akceptacji jest w tym

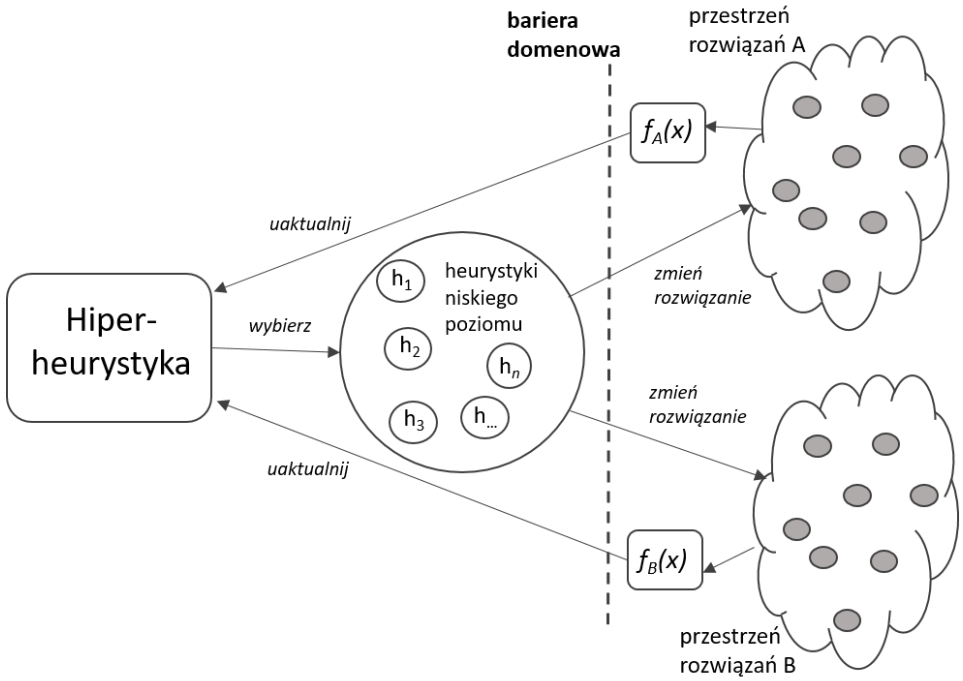
przypadku inny, nazywa się MONTECARLO. Jeśli rozwiązanie jest lepsze niż dotychczas znalezione, to zawsze jest akceptowane. W przeciwnym wypadku może zostać odrzucone z pewnym prawdopodobieństwem zależnym od czasu dotychczasowych obliczeń $\exp(\Delta/\delta(t))$, gdzie Δ przedstawia różnicę funkcji celu nowego i poprzedniego rozwiązania, natomiast temperatura $\delta(t)$ jest zdefiniowana jako

$$\delta(t) = \beta \, log^{-1}(\lfloor t/m \rfloor + 2)$$

gdzie *t* jest to czas który upłynął od początku, a *m* jest liczbą iteracji, w czasie których temperatura jest stała w procesie wychładzania. Interpretacja mechanizmu MONTECARLO jest następująca: w miarę upływu czasu temperatura wychładza się i hiper-heurystyka jest mniej skłonna do akceptowania heurystyk pogarszających rozwiązanie.

Wszystkie hiper-heurystyki były testowane dla dwóch zbiorów heurystyk niskiego poziomu: pierwszego, który oferował tylko proste kroki, np. wstawienie *k*-meru do rozwiązania, przesunięcie lub usunięcie *k*-meru; drugi zbiór obejmował dodatkowo operacje na kilku sąsiadujących ze sobą w rozwiązaniu *k*-merach, które były względem siebie przesunięte nie więcej niż *l* nukleotydów. Wyniki przeprowadzonych testów eksperymentalnych pokazały, że bardzo ważne jest zaimplementowanie dobrego algorytmu uczącego, (ROULETTEC często nie potrafił znaleźć dobrego rozwiązania). Istotny wpływ na wynik rozwiązania miał także właściwy dobór heurystyk niskiego poziomu. Dla dobrze dobranego zbioru heurystyk wszystkie algorytmy potrafiły znaleźć dobre rozwiązanie, natomiast przy złym wyborze przesunięcia *l* tylko SIMANN oraz RANKEDC osiągały dobry rezultat. Dla testów na rzeczywistych instancjach ponownie SIMANN okazał się najlepszym algorytmem razem z ROULETTEC. Dobry, aczkolwiek niespodziewany wynik metody, która działa na zasadzie ruletki spowodowany był faktem, iż większość algorytmów wpadała w lokalne optimum, a ROULETTEC burzył aktualne rozwiązanie i był w stanie przeskoczyć w inne miejsce przestrzeni rozwiązań.

W kolejnych badaniach nad hiper-heurystykami - [P4], zaproponowany został nowy model hiper-heurystyki wykorzystujący jednorodny sposób kodowania rozwiązania dla różnych problemów kombinatorycznych. Pozwoliło to na zdefiniowanie jednego zbioru heurystyk niskiego poziomu, które mogły rozwiązywać problemy z różnych dziedzin. Heurystyki są teraz niezależne od problemu, zatem bariera domenowa jest przeniesiona poniżej zbioru heurystyk (Rys. 4). W standardowym podejściu mając zbiór heurystyk niskiego poziomu specyficznych dla danego problemu oraz funkcję oceny można rozwiązać problem *P* używając jednej z istniejących hiper-heurystyk. W zaproponowanym podejściu mając zakodowane rozwiązanie oraz funkcję oceny rozwiązania możemy rozwiązać problem *P* wykorzystując istniejącą hiper-heurystykę oraz zbiór ogólnych heurystyk niskiego poziomu operujących na zdefiniowanym wcześniej kodowaniu rozwiązania, co znacząco może zmniejszyć nakład pracy. Jako hiper-heurystyki wykorzystane zostały algorytmy zaproponowane w pracy [P7]: RANKEDC, DECOMPC, STRAIGHTC, ROULETTEC, TABUS i SIMANN.



Rys. 4. Schemat przedstawiający przeniesienie bariery domenowej poza obszar heurystyk niskiego poziomu, co było możliwe dzięki jednorodnemu kodowaniu rozwiązań różnych problemów.

W zaproponowanym podejściu rozwiązanie jest reprezentowane jako sekwencja *S* unikalnych liczb od 1 do *n*. Każda instancja jest reprezentowana przez pełny graf skierowany *G* z *n* wierzchołkami i łukami z wagami. $D_{i,j}$ oznacza wagę łuku pomiędzy wierzchołkiem *i* oraz *j*. Graf nie musi być symetryczny, czyli $D_{i,j}$ może być różne od $D_{j,i}$. Z każdym wierzchołkiem skojarzona jest nagroda Pr_i oraz kara Pe_i. Dodatkowo jest zdefiniowane ograniczenie *Q*. Rozpatrywano 5 różnych problemów:

- Sekwencjonowanie przez hybrydyzację. Wagi $D_{i,j}$ zdefiniowane są jako przesunięcie pomiędzy *k*-merami *i* oraz *j*. *Q* jest maksymalną długością ścieżki. Wartości Pr_i oraz Pe_i ustawione są na 0 jako nieużywane. Problem jest zdefiniowany następująco:

  $$\text{maksymalizuj } |S|$$

  Przy ograniczeniu:

  $$\sum_{i=1}^{|S|-1} D_{S_i,S_{i+1}} \leq Q$$

  Ze względu na to iż funkcja celu nie rozróżniała rozwiązań z taką samą liczbą elementów ale różniącą się długością ścieżki (czyli długością sekwencji DNA) wprowadzono alternatywną funkcję celu:

  $$\text{maksymalizuj } (|S|+1) \times factor - length$$

  gdzie *factor* jest liczbą znacząco większą niż długość ścieżki, a *length* jest długością ścieżki wyrażoną jako lewa część ograniczenia.

- Problem komiwojażera. Wagi $D_{i,j}$ przechowują odległości pomiędzy miastami, a pozostałe zmienne ustawione są na 0. Problem jest sformułowany jako:

  $$\text{minimalizuj} \sum_{i=1}^{|S|} D_{S_i,S_{(i \bmod n)+1}} \leq Q$$

  przy ograniczeniu: Rozwiązanie musi zawierać dokładnie *n* elementów $|S| = n$

- Problem komiwojażera z wąskim gardłem. Zmienne są zdefiniowane w taki sam sposób jak w problemie komiwojażera. Natomiast sam problem jest zdefiniowany:

$$\text{minimalizuj} \max_{1 \le i \le |S|} D_{S_i, S_{(i \bmod n)+1}}$$

przy ograniczeniu: Rozwiązanie musi zawierać dokładnie *n* elementów $|S| = n$

- Problem komiwojażera z nagrodami. Ten problem w pełni wykorzystuje wszystkie zmienne przewidziane dla reprezentacji problemu. Zadaniem jest:

$$\text{minimalizuj} \sum_{i=1}^{|S|} D_{S_i, S_{(i \bmod |S|)+1}} + \sum_{v \notin S} Pe_v$$

przy ograniczeniu:

$$\sum_{v \notin S} Pr_v \ge Q$$

- Problem plecakowy. Pr_i wykorzystywane jest do przechowania wartości elementów, natomiast Pe_i do przechowania ich wagi. Ograniczenie *Q* jest maksymalną dopuszczalną wagą elementów w plecaku. Macierz *D* nie jest wykorzystywana w tym problemie. Rozwiązanie *S* przechowuje elementy w plecaku. Zadaniem jest:

$$\text{maksymalizuj} \sum_{i=1}^{|S|} Pr_{S_i}$$

przy ograniczeniu:

$$\sum_{i=1}^{|S|} Pe_{S_i} \le Q$$

Zbiór heurystyk niskiego poziomu zaproponowanych dla zdefiniowanego powyżej jednorodnego kodowania rozwiązania składał się z prostych heurystyk, np. wstawienie elementu do rozwiązania, usunięcie, przesunięcie elementu, przesunięcie lub odwrócenie kolejności kilku kolejnych elementów w rozwiązaniu, oraz usunięcie największej odległości pomiędzy sąsiadującymi elementami w rozwiązaniu. Warto zwrócić uwagę na to, iż niektóre heurystyki są zupełnie bezużyteczne dla testowanych problemów, np. ostatnia z heurystyk nie znajdzie największego połączenia pomiędzy sąsiadującymi elementami w problemie plecakowym, gdyż macierz *D* nie jest w ogóle wykorzystywana. Przeprowadzone testy eksperymentalne pokazały, że rzeczywiście większość hiper-heurystyk nauczyła się, które heurystyki są nieużyteczne dla rozwiązywanego problemu i nie wykorzystywała ich w trakcie obliczeń. Dla trzech spośród testowanych problemów uzyskane wyniki były bliskie optymalnym, przy czym najlepsze rezultaty osiągnęły hiper-heurystyki RANKED, DECOMPC oraz SIMMANN. Dla problemów komiwojażera z nagrodami oraz z wąskim gardłem niestety nie udało się uzyskać satysfakcjonujących wyników w trakcie założonego czasu. Może to wynikać z faktu, iż funkcja celu nie jest ciągła i jest wiele rozwiązań z taką samą wartością funkcji, które są trudno rozróżnialne (komiwojażer z wąskim gardłem) lub też zbiór testowanych heurystyk niskiego poziomu był niewystarczający do przeszukania większej części przestrzeni rozwiązań (komiwojażer z nagrodami). Jednakże uzyskane wyniki są bardzo obiecujące i być może przy wykorzystaniu innego zbioru heurystyk uda się uzyskać lepszy rezultat dla wszystkich testowanych problemów, a także dla innych problemów kombinatorycznych, których rozwiązanie będzie można zakodować w powyżej przedstawiony sposób.

4.3.4. Metody dla problemu asemblacji

Asemblacja DNA to proces składania krótkich sekwencji w dłuższe fragmenty. Sam proces składania podobny jest do problemu sekwencjonowania przez hybrydyzację, z tym że różni się skalą problemu.

W SBH do złożenia jest kilkaset krótkich sekwencji o długości od kilku do kilkunastu nukleotydów, natomiast w procesie asemblacji każda z sekwencji składa się z kilkuset nukleotydów, a liczba sekwencji, zależna od długości badanego genomu i głębokości pokrycia, może sięgać nawet miliardów. Sekwencje mogą również zawierać błędy kilku nukleotydowe, np. wstawienia dodatkowych liter, brak litery, czy zamiana na inną literę. Dodatkowo sekwencje mogą pochodzić z obu nici helisy DNA, więc trzeba zduplikować ich liczbę o sekwencje odwrotnie komplementarne, a w trakcie rekonstrukcji wykorzystać tylko jedną z pary sekwencji.

W literaturze wykorzystywane są najczęściej dwa podejścia do rozwiązania problemu asemblacji: grafy nałożeń OLC (*overlap layout consensus*) oraz grafy dekompozycji DBG (*decomposition graphs, de Bruijn graphs*).

Grafy nałożeń, podobnie jak grafy Lysova dla SBH w wierzchołkach przechowują sekwencje. Łuki łączą wierzchołki, których sekwencje nachodzą na siebie, przy czym dopuszcza się niedokładne dopasowanie ze względu na błędy w sekwencjach. Minimalna długość zachodzenia na siebie sekwencji oraz dopuszczalna liczba błędów jest często sparametryzowana. Informacja o długości nakładania się sekwencji i liczbie błędów w dopasowaniu jest przechowywana w łuku i jest wykorzystywana do wyboru właściwej ścieżki. Podobnie jak w modelu Lysova dla SBH, w grafie nałożeń moglibyśmy szukać ścieżki Hamiltona, jednak ze względu na wagi na łukach musielibyśmy szukać ścieżki o najmniejszej liczbie błędów lub o największym zysku. W praktyce jednak brak równomiernego pokrycia fragmentami DNA badanej sekwencji nie pozwala na znalezienie jednej ścieżki. Ponadto powtarzające się fragmenty sekwencji genomu powodują że w grafie są rozgałęzienia i trudno jednoznacznie wybrać właściwą ścieżkę. W efekcie uzyskujemy wiele rozłącznych ścieżek, które następnie zamieniane są na sekwencje konsensusowe, tzw kontigi, które są złożeniem sekwencji uporządkowanych przez ścieżkę.

W *grafach dekompozycji*, podobnych do grafów Pevznera z SBH, każda sekwencja dekomponowana jest na *k*-mery, które są przechowywane w łukach (w wierzchołkach mamy zatem (*k*-1)-mery będące sufiksem i prefiksem *k*-merów). Każda sekwencja jest zatem ścieżką w grafie przechodzącą przez kolejne *k*-mery przesunięte względem siebie o jeden nukleotyd. W pierwszej fazie graf jest naprawiany w celu usunięcia rzadko występujących *k*-merów, które są najczęściej efektem błędów w sekwencjach. Następnie w grafie poszukiwane są ścieżki zawierające wszystkie łuki. Każda ścieżka jest automatycznie zamieniana na kontig. Również w tym podejściu bardzo ważnym elementem jest wychwycenie sytuacji, w której ścieżka w grafie ulega rozgałęzieniu, najczęściej na skutek powtórzeń w badanych sekwencjach genomu. Nie wykrycie rozgałęzienia i kontynuacja przechodzenia ścieżką może się wiązać z niepoprawnym odtworzeniem sekwencji genomu.

Różne modele i algorytmy dla problemu asemblacji zostały szerzej opisane i porównane przeze mnie oraz współautorów moich prac w artykułach [P3] oraz [P9]. Oba podejścia mają swoje wady i zalety. Wadą podejścia OLC jest duża złożoność pamięciowa w celu przechowywania grafu nałożeń w pamięci, oraz złożoność czasowa wyznaczenia nakładania się sekwencji. Natomiast wadą podejścia DBG jest utrata informacji podczas dekomponowania odczytów na *k*-mery i większa w związku z tym wrażliwość na powtórzenia krótkich fragmentów DNA w sekwencji.

Poniżej opisane zostaną trzy metody opracowane przeze mnie wraz z zespołem, wykorzystujące podejście OLC: zmodyfikowany graf DNA [P8], graf acykliczny [P6] oraz hybrydowy algorytm z grafem nałożeń, w którym dopasowania wyznaczane są przy pomocy obliczeń na kartach graficznych [P2].

Pierwsza metoda [P8] opiera się na zmodyfikowanym grafie DNA zaproponowanym przez Lysova dla SBH. Ze względu na dużą liczbę sekwencji wejściowych i niewykonalne czasowo zadanie porównania wszystkich par sekwencji ze sobą, w trakcie budowy grafu nałożeń, zastosowano heurystykę, która pozwoli na stosunkowo szybkie znalezienie par sekwencji *obiecujących*, które z dużym prawdopodobieństwem będą się ze sobą nakładać. Heurystyka ta wyznacza dla każdej sekwencji zbiór okien, czyli kilkunukleotydowych fragmentów, przesuniętych względem siebie o 1. Załóżmy że wyznaczane jest prawdopodobieństwo nakładania się ze sobą sekwencji A oraz B. Oznaczmy jako W_A

zbiór wszystkich okien w sekwencji A. W_B będzie zbiorem wszystkich okien ze zbioru W_A, które są jednocześnie w B. Następnie sprawdzamy, które okna występują w takiej samej kolejności dla A i B:

$$W_{AB} = \{s \in W_B : pos_A(pred_B(s)) < pos_A(s)\}$$

gdzie $pos_A(s)$ jest pozycją, w której okno s zaczyna się w sekwencji A, $pred_B(s)$ jest oknem poprzedzającym s w sekwencji B. Wówczas miara p ocenia, która z par sekwencji oznaczona będzie jako obiecująca

$$p = \frac{|W_{AB}| + 1}{|W_B|}$$

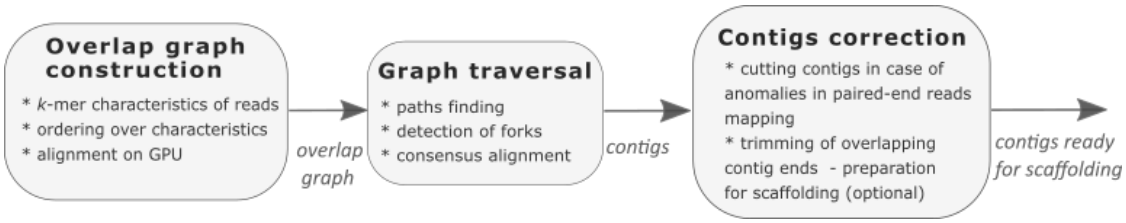
Dla wszystkich par obiecujących wyznaczone jest dokładne dopasowanie algorytmem Smitha-Watermana i jeśli jest ono akceptowalne, dodawany jest łuk w grafie między wierzchołkami A i B. Dalsza część algorytmu opiera się w dużej mierze na strategii dla modelu OLC. Ze względu na duże zużycie pamięci oraz czasochłonne obliczenia, przy użyciu tego algorytmu udało się zasemblować genom bakterii o długości ok 1,8 milionów nukleotydów, na podstawie zbioru złożonego z 300 tys. sekwencji.

W kolejnej pracy [P6] zaproponowałam wraz zespołem inny algorytm dla problemu asemblacji *de novo*, który transformuje graf nałożeń, do grafu acyklicznego, a następnie szuka w takim grafie ścieżek. Problem znalezienia rozłącznych ścieżek jest podobny do problemu minimalnego pokrycia ścieżkami, który jest w ogólnym przypadku NP-trudny, jednak w przypadku grafu acyklicznego – łatwy obliczeniowo. Jednakże, aby wykorzystać zalety problemu minimalnego pokrycia ścieżkami dla grafu skierowanego, acyklicznego (*directed acyclic graph*) należy uporać się z dwiema trudnościami: (i) graf nałożeń zawiera cykle oraz (ii) każdy wierzchołek w grafie jest podwójny (składa się z sekwencji oryginalnej oraz drugiej, odwrotnie komplementarnej). Jeśli wyznaczone byłoby dopasowanie każdej pary sekwencji, to graf byłby pełny. Jednak nie jest to konieczne (oraz nie wykonalne czasowo ze względu na dużą liczbę sekwencji), gdyż wiadomo, że nie wszystkie sekwencje nakładają się na siebie. Przy pomocy heurystyki wybierane są pary sekwencji do porównania, i jeśli wyznaczone dopasowanie sekwencji nie przekracza dopuszczalnego progu na liczbę błędów, taki łuk jest dodawany do grafu nałożeń. Następnie wszystkie silnie spójne składowe grafu są oddzielnie transformowane do acyklicznej formy.

W celu rozwiązania problemu (i), czyli przerwania cykli w ściśle spójnych składowych grafu, zaproponowane zostały dwie heurystyki: pierwsza usuwa ze składowej grafu najgorszy łuk, tak długo dopóki występuje cykl. Druga, bardziej skomplikowana heurystyka dla każdej spójnej składowej grafu wybiera wierzchołek, który dzielony jest dwa, v_s połączony ze wszystkimi łukami wychodzącymi z wybranego wierzchołka, oraz v_t połączony ze wszystkimi wchodzącymi wierzchołkami. Następnie wykorzystywany jest algorytm wyznaczania maksymalnego przepływu (max-flow min-cut) w celu znalezienia najmniejszego przecięcia potrzebnego do rozłączenia źródła v_s od ujścia v_t. W podobny sposób (przy użyciu algorytmu wyznaczania maksymalnego przepływu) rozwiązywany jest problem podwójnych wierzchołków (ii). Dla każdej pary wierzchołków, dla której istnieje łuk, istnieje również łuk skierowany w drugą stronę dla ich odwrotnie komplementarnych odpowiedników. Ponieważ w trakcie przechodzenia grafu możemy wykorzystać tylko jedną sekwencję z wierzchołka, więc będzie można wykorzystać również tylko jeden łuk. Po transformacji grafu nałożeń do grafu acyklicznego algorytm minimalnego pokrycia ścieżkami wyznaczał ścieżki od najdłuższej do najkrótszej. Eksperyment obliczeniowy przeprowadzony dla zestawu danych z sekwencjonowania genomu bakterii z pracy [P8] pozwolił na uzyskanie długich kontigów często dłuższych niż w przypadku innych metod.

Ze względu na to, iż metody zaproponowane w pracach [P8] oraz [P6] działały tylko na mniejszych zestawach danych, np. dla genomu bakterii, gdy liczba sekwencji nie przekraczała kilkuset tysięcy, rozpoczęliśmy wraz z zespołem prace nad asemblerem, który w efektywny sposób przetworzy wielomilionowe zbiory sekwencji, w niedługim czasie wyznaczy dopasowania par sekwencji i znajdzie kontigi, które będą reprezentowały badany genom (to znaczy będzie można dopasować je w całości do

genomu). Prowadziliśmy badania nad wstępnym przetwarzaniem danych z sekwencjonowania w celu zmniejszenia zbiorów wejściowych i kompresji sekwencji [P5], co zostanie szerzej opisane w sekcji 4.3.5. Kilkuletnia praca nad nowym algorytmem zaowocowała narzędziem GRASShopPER opublikowanym w [P2]. Asembler korzysta z podejścia OLC, podobnie jak poprzednie dwie metody. Ogólny schemat działania przedstawia Rys. 5.



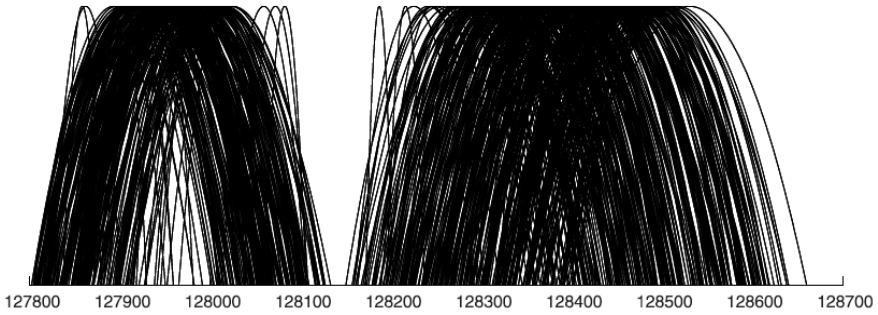
Rys. 5. Ogólny schemat działania assemblera GRASShopPER.

W pierwszej części tworzony jest graf nałożeń. Pary sekwencji, które są do siebie podobne, wybierane są dzięki charakterystykom *k*-merowym. Z każdego odczytu wyekstrahowane są wszystkie podsekwencje o długości *k*, a następnie zliczana jest liczba wystąpień każdego *k*-meru. Charakterystyki *k*-merowe sortowane są w pierwszej kolejności względem liczności *k*-merów, a następnie leksykograficznie. Sekwencje, których charakterystyki leżą blisko siebie po posortowaniu, sprawdzane są dalej algorytmem Needelmana-Wunscha (NW) do wyznaczania optymalnego dopasowania. Implementacja algorytmu NW została zaimplementowana na kartach graficznych [FKB+13]. Jeśli wynik dopasowania pary sekwencji jest dopuszczalny (pod uwagę brane są przesunięcie sekwencji względem siebie oraz liczba błędów) dodawany jest łuk pomiędzy odpowiadającą im parą wierzchołków w grafie nałożeń. Podczas tworzenia grafu wykonywanych jest wiele dodatkowych kroków, aby wstawione zostały wszystkie łuki, dla których sekwencje nakładają się na siebie (dokładna metoda szerzej opisana w [P2]).

W następnej części metody następuje trawers grafu ze szczególnym uwzględnieniem rozgałęzień w grafie. Rozgałęzienia występują w grafie najczęściej na skutek powtórzeń w genomie lub z powodu błędów w sekwencjach wejściowych. Aby poprawnie zrekonstruować kontigi należałoby zakończyć aktualnie tworzoną ścieżkę w grafie w momencie, gdy napotkane zostanie rozgałęzienie. W celu poprawnej rekonstrukcji badanego genomu zdefiniowaliśmy stan $S = s_1, s_2, \ldots, s_r$ składający się z *r* ostatnio odwiedzonych wierzchołków s_i, oraz listę kandydatów *C*, będącymi następnikami wierzchołków z *S*. $C = \bigcup_{i=1}^{r} out(s_i)$, gdzie $out(s_i)$ jest zbiorem następników s_i. Każdy wierzchołek-kandydat oceniany jest poprzez ważoną sumę swoich poprzedników ze stanu (wierzchołek ostatnio dodany do *S* ma największą wagę w ocenie). Kandydat z największą liczbą punktów zostaje wybrany do stanu, wyrzucając jednocześnie z niego wierzchołek, który największą liczbę iteracji przebywał w *S*. Rozgałęzienie rozpoznawane jest w grafie, gdy po usunięciu z *S* wierzchołka tracona jest jednocześnie duża liczba kandydatów. Dzieje się tak w przypadku, gdy wszystkie wierzchołki ze stanu przesunęły się na jedną z gałęzi, a usunięcie z *S* wierzchołka, który jako jedyny miał następniki na innej gałęzi grafu spowodowało utratę połączenia z wieloma kandydatami z *C*. W momencie stwierdzenia przez algorytm istnienia rozgałęzienia, aktualna ścieżka zostaje zakończona ostatnio usuniętym wierzchołkiem ze stanu. Po zakończeniu trawersowania grafu każda ścieżka zmieniana jest na sekwencję konsensusową, czyli kontig.

Chociaż graf przeglądany jest z wielką uwagą, niestety nie wszystkie rozgałęzienia w grafie zostaną znalezione. W trzecim etapie (por. Rys 5) wykorzystywana jest zatem dodatkowa informacja ze sparowanych sekwencji (w trakcie sekwencjonowania możliwe jest odczytanie pary sekwencji oddalonych od siebie o około kilkaset nukleotydów). Pary sekwencji, które w pierwszej części metody traktowane były niezależnie, zostają zmapowane do kontigów i sprawdzamy czy znajdują się one w poprawnej odległości od siebie. Możemy rozróżnić dwa typy anomalii w mapowaniu par: (i) niespójność w mapowaniu się sparowanych sekwencji (por. Rys 6), (ii) znaczne przekroczenie dopuszczalnej liczby par sekwencji, z których jedna mapuje się do sprawdzanego kontigu, a druga z

pary do innego kontigu. Oba przypadki świadczą o niewykrytym rozgałęzieniu, a w celu znalezienia obu typów anomalii zaproponowałam algorytm hiper-heurystyczny. Hiper-heurystyka jest algorytmem zachłannym, który działa na dwóch prostych heurystykach niskiego poziomu. Pierwsza z nich wykrywa przerwę między sparowanymi odczytami, które kolejno są zmapowane do kontigu (i). Druga szuka pary sekwencji, które są zmapowane w dużej odległości od siebie lub na innym kontigu (ii). W przypadku wykrycia rozgałęzienia kontig dzielony jest na dwie części.



Rys. 6. Przykład anomalii w mapowaniu sparowanych sekwencji do fragmentu kontigu

Testy obliczeniowe zostały przeprowadzone na 3 zestawach danych rzeczywistych, różniących się długością badanego genomu, liczbą sekwencji wejściowych, pokryciem genomu oraz charakterystyką powtórzeń. Porównaliśmy kontigi wynikowe dla różnych asemblerów przy wykorzystaniu szeroko-rozpowszechnionego narzędzia QUAST. Biorąc pod uwagę różne miary uzyskaliśmy wyniki porównywalne z najlepszymi asemblerami dostępnymi na rynku. Nasz algorytm tworzył kontigi, które procentowo najbardziej pokrywały badane genomy. Uzyskane przez GRASShopPERa kontigi były bardzo dobrej jakości, co sprawdzaliśmy mapując każdy z nich do badanego genomu referencyjnego. Niektóre asemblery: Velvet, SPAdes produkowały dłuższe kontigi, jednak ich jakość często była gorsza, a suma długości źle zasemblowanych kontigów przekraczała 1% długości genomu, co czyni taki asembler niepraktycznym.

Problem asemblacji *de novo* jest znacznie bardziej skomplikowany niż SBH, głównie ze względu na rozmiar instancji wejściowych, dlatego metody rozwiązujące ten problem są specjalnie dla niego dedykowane. Metody ogólnego działania, typu hiper-heurystyki, nie sprawdziłyby się w praktyce ze względu na zbyt dużą przestrzeń rozwiązań (np. ułożenie kilkudziesięciu milionów sekwencji w odpowiednim porządku). Hiper-heurystyka, która nie znałaby charakterystyki rozwiązania i korzystała tylko z prostych heurystyk niskiego poziomu mogłaby nie znaleźć rozwiązania bliskiego optymalnemu. Dla problemu asemblacji zaproponowałam dwie hiper-heurystyki, które rozwiązują jeden z etapów złożonego problemu asemblacji *de novo*. Pierwsza z nich, opisana powyżej oraz w [P2] i zaimplementowana w zaproponowanym przez nas asemblerze, wyszukiwała niewykrytych na wcześniejszym etapie rozgałęzień. Druga, zaproponowana w [P1], również dotyczy wyszukiwania rozgałęzień, lecz na wcześniejszym etapie – przechodzenia grafu. Hiper-heurystyka potrafiłaby wykryć rozgałęzienie np. na podstawie zmiany pokrycia. Badania musiałyby być poprzedzone trenowaniem offline na przygotowanych zestawach testowych w celu nauki rozpoznawania specyfiki budowy grafu w pobliżu rozgałęzienia. Heurystyki niskiego poziomu mogłyby być prostymi krokami typu ‘zakończ ścieżkę’ oraz ‘kontynuuj przechodzenie ścieżki’, a także heurystyki wydłużające ścieżkę w obie strony. Jednak ze względu na koszt pomyłki skręcenia w złą ścieżkę w grafie i możliwość niepoprawnego złożenia kontigu, po wstępnych testach zdecydowaliśmy się na specjalną strategię wykrywania rozgałęzień w asemblerze z [P2].

4.3.5. Kompresja danych i transfer

Ostatnie zagadnienie, którym zajmowałam się w prezentowanym cyklu habilitacyjnym, związane jest ze wstępnym przetwarzaniem i kompresją danych pochodzących z sekwenatora i transferem ich na dedykowany serwer zewnętrzny. Zagadnienie to było jednym z zadań w kierowanym przeze mnie grancie Narodowego Centrum Nauki „Wysoko wydajne obliczenia dla sekwencjonowania DNA nowej

generacji". W Europejskim Centrum Bioinformatyki i Genomiki zlokalizowanym na terenie Instytutu Informatyki Politechniki Poznańskiej jesteśmy w posiadaniu sekwenatora Illuminy Genome Analyzer IIx. Z uwagi na szybko rosnącą ilość danych, dość szybko okazało się konieczne składowanie wyników sekwencjonowania na zewnętrznym serwerze. W pracy [P5] przedstawiliśmy wyniki optymalizacji sposobu kompresji danych z surowych eksperymentów oraz porównanie szybkości transferu danych na serwer w PLATON-U4 przy wykorzystaniu protokołów SFTP oraz GridFTP. Tamże, zaproponowaliśmy również alternatywny sposób wstępnej obróbki (*preprocessing*) danych z sekwencjonowania przed asemblacją *de novo*. Sekwencja operacji które należy wykonać składała się z kroków:

a) odrzucenie sekwencji z niezidentyfikowanymi nukleotydami (N)
b) dodanie odczytów odwrotnie komplementarnych
c) kompresja danych – każdy nukleotyd zapisany jest na dwóch bitach
d) sortowanie leksykograficzne i usunięcie duplikatów
e) utworzenie Δ-sekwencji i ponowne sortowanie. Δ-sekwencje są wariantami sekwencji z dopuszczalnym przesunięciem Δ. Dla jednej sekwencji można skonstruować Δ+1 jej Δ-sekwencji, które są sufiksami rozpoczynającymi się od *i*-tego nukleotydu, $i \in \langle 1, \Delta + 1 \rangle$
f) analiza nakładania się sekwencji – para Δ-sekwencji, dla których jeden element jest prefiksem drugiego z przesunięciem=0 odpowiada parze sekwencji nakładających się na siebie z przesunięciem między 1 a Δ. Implementacja sortowania na kartach GPU z wykorzystaniem algorytmu SRTS.
g) wybór par obiecujących i wstępne łączenie kontigów, dla których nie wykryto rozgałęzienia

Wykorzystanie powyższych kroków na dwóch zestawach danych rzeczywistych dla genomów *E. coli* oraz *C. elegans* pozwoliło zredukować dane wejściowe dla problemu asemblacji do odpowiednio 14% i 27% początkowej liczby sekwencji (dla Δ=10). Pomimo tak obiecujących wyników, nie udało się włączyć tejże procedury do narzędzia GRASShopPER, głównie ze względu na utratę informacji o sekwencjach sparowanych.

4.3.6. Podsumowanie

Badania zaprezentowane w niniejszej sekcji, wykorzystując metody informatyczne, wnoszą istotny wkład w dziedzinę biologii obliczeniowej i bioinformatyki w kontekście poznawania genomów. O ile odczytywanie genomu w małej skali (sekwencjonowanie przez hybrydyzację) dla niewielkich instancji mogłoby zostać zrealizowane „ręcznie", to już sekwencjonowanie w dużej skali niestety nie, z uwagi na rozmiar instancji i dodatkowe trudności w problemie asemblacji *de novo*. Badania obejmowały szeroką analizę metod hiper-heurystycznych nie tylko w kontekście rozwiązywania problemu SBH, lecz także zaowocowały zaproponowaniem nowego podejścia do hiper-heurystyk z jednorodnym kodowaniem, co umożliwia na wykorzystanie zarówno jednej hiper-heurystyki, jak i zbioru heurystyk niskiego poziomu do rozwiązania kilku problemów kombinatorycznych [P1,P4,P7]. Dokonano również usystematyzowania wiedzy dotyczącej grafów wykorzystywanych do modelowania i rozwiązywania problemów sekwencjonowania w małej i dużej skali [P3,P9]. Zaproponowano różne modele i metody do asemblacji *de novo* [P1,P2,P6,P8] oraz wstępnego przetwarzania danych z sekwencjonowania [P5]. Rozpoznawalność zespołu w kontekście procesu czytania genomów (głównie dla rozwiązywania problemu asemblacji *de novo*) zaowocowała uczestnictwem w projekcie genomicznej mapy Polaka, który ma na celu stworzenie bazy z różnicami charakterystycznymi dla populacji Polaków, a także grup etnicznych mieszkających na terenie Polski. Jednym z zadań jest też utworzenie genomu referencyjnego na bazie zsekwencjowanych genomów, co w efekcie może przynieść korzyść w postaci wypełnienia luk istniejących w obecnej wersji genomu referencyjnego w bazie GenBank.

[5] Omówienie pozostałych osiągnięć naukowo-badawczych

5.1. Lista prac – pozostałe osiągnięcia

Numeracja listy publikacji jest zgodna z Załącznikiem 3 – wykaz dorobku naukowego.

[A1] M. Goralski, P. Sobieszczanska, A. Obrepalska-Steplowska, A. Swiercz, A. Zmienko, M. Figlerowicz, "A gene expression microarray for Nicotiana benthamiana based on de novo transcriptome sequence assembly" *Plant Methods*, 12:28, **2016**, doi:10.1186/s13007-016-0128-4. **Pkt MNiSW(2016)=40; IF(2016)=3.510**

[A2] K. Klonowska, L. Handschuh, A. Swiercz, M. Figlerowicz, P. Kozlowski, "MTTE: an innovative strategy for the evaluation of targeted/exome enrichment efficiency", *Oncotarget* 7, **2016**, pp.67266-67276, **Pkt MNiSW(2016)=35; IF(2016)=5.168**

[A3] D. Santoni, A. Świercz, A. Żmienko, M. Kasprzak, M. Błażewicz, P. Bertolazzi, G. Felici, "An integrated approach (CLuster Analysis Integration Method) to combine expression data and protein-protein interaction networks in agrigenomics: Application on Arabidopsis thaliana." *OMICS: A Journal of Integrative Biology* 2, **2014**, pp.155-165. **Pkt MNiSW(2014)=30; IF(2014)=2.362**

[A4] D. Kowalczykiewicz, A Swiercz, L. Handschuh, K. Lesniak, M. Figlerowicz, J. Wrzesinski, "Characterization of *Sus scrofa* small non-coding RNAs present in both female and male gonads", *PLOS ONE*,9(11): e113249, **2014**. **Pkt MNiSW(2014)=40; IF(2014)=3.234**

[A5] J. Blazewicz, B. Bosak, P. Gawron, M. Kasprzak, K. Kurowski, T. Piontek, A. Swiercz, "Highly efficient parallel approach to the next-generation DNA sequencing", in Proceedings of PPAM'11, *Lecture Notes in Computer Science* 7204, **2012**, pp.262-271.

[A6] J. Błażewicz, F. Glover, M. Kasprzak, W.T. Markiewicz, C. Oğuz, D. Rebholz-Schuhmann, A. Świercz "Dealing with repetitions in sequencing by hybridization" *Computational Biology and Chemistry* 30, no. 5, **2006**, pp. 313-320. **Pkt MNiSW(2010)=32; IF(2006)=2.135**

[A7] J. Błażewicz, C. Oğuz, A. Świercz, J. Węglarz "DNA sequencing by hybridization via genetic search" *Operations Research* 54, no. 6, **2006**, pp. 1185-1192. **Pkt MNiSW(2010)=32; IF(2006)=1.234**

[A8] J. Błażewicz, P. Formanowicz, M. Kasprzak, W. T. Markiewicz, A. Świercz "Tabu search algorithm for DNA sequencing by hybridization with isothermic libraries" *Computational Biology and Chemistry* 28, **2004**, pp. 11-19. **Pkt MNiSW(2010)=32; IF(2004)=1.655**

[C1] Zgłoszenie patentowego wynalazku w Urzędzie Patentowym Rzeczypospolitej Polskiej w dniu 26.01.2017, nr zgłoszenia **P. 420318**. *„Model komórkowy ludzkiego raka jajnika w hodowli o zaindukowanej paklitakselem odwrotnej oporności na paklitaksel i cisplatynę oraz zastosowanie tego modelu”*.

5.2. Omówienie pozostałych osiągnięć

Badania przedstawione w punktach 4.3.2 - 4.3.5 były realizowane równolegle z projektami, w których między innymi przeprowadzałam analizę bioinformatyczną danych pochodzących z sekwencjonowania. W trakcie realizacji zadań w projektach zdobywałam wiedzę na temat specyfiki danych, błędów charakterystycznych dla różnych sekwenatorów oraz różnic w sekwencjach genomów organizmów (np. różnica w ilości i typie występujących powtórzeń). Pozwoliło to na dokładniejsze zaprojektowanie algorytmu asemblacji, a także na określenie że nie jest możliwe 100% dopasowanie kontigów do genomu referencyjnego na przykład ze względu na różnice na pojedynczych pozycjach (tzw. *SNP*, *Single Nucleotide Polymorphism*), oraz różnice pomiędzy sekwencjami dwóch kopii tego samego chromosomu w jednym organizmie. Projekty w których brałam udział i zakończyły się powstaniem publikacji dotyczyły:

- [A4] analizy różnic między niekodującymi krótkimi sekwencjami RNA w gonadach męskich i żeńskich organizmu *Sus scrofa*

- [A3] integracji danych ekspresji genów pochodzących z eksperymentów na mikromacierzach lub z sekwencjonowania transkryptomów oraz danych z bazy interakcji białkowych (Protein-Protein Interaction network) dla organizmu *Arabidopsis thaliana*
- [A2] oceny efektywności wzbogacania próbek przy sekwencjonowaniu egzomów na przykładzie organizmu *Homo sapiens*
- [A1] zaprojektowania sond na mikromacierzy dla organizmu *Nicotiana benthamiana* na podstawie danych z sekwencjonownia transkryptomów

Oprócz tego wykonywałam analizę bioinformatyczną badania ekspresji różnicowej dla hodowli komórek rakowych traktowanych różnymi stężeniami środków chemicznych, analizowałam wyniki oraz zaproponowałam wybór najciekawszych genów z różnymi wzorcami ekspresji. W styczniu 2017 dokonaliśmy zgłoszenia patentowego [C1], a obecnie trwają prace nad publikacją oraz komercjalizacją hodowli komórek rakowych opornych na działanie leków.

Ponadto brałam udział w projektach zajmujących się znajdowaniem fragmentów z różną liczbą kopii (CNV) dla genomu jedwabnika, badaniem różnicowej ekspresji krótkich RNA dla genomu ziemniaka zainfekowanego wirusem Y (praca doktorska dr Karoliny Morgiewicz, IBB PAN), badaniem ekspresji różnicowej komórek człowieka ze szczególnym uwzględnieniem białek PUMILIO i MAELSTROM (publikacja w przygotowaniu). Obecnie nadzoruję nad pracą studentów, którzy wykorzystają narzędzie GRASShopPER w celu znalezienia różnych wariantów strukturalnych *de novo*.

Moje doświadczenie z obróbką bioinformatyczną danych z eksperymentów z udziałem sekwenatora oraz mikromacierzy wykorzystałam do przygotowania autorskich cyklów wykładów i zajęć laboratoryjnych „Analiza danych wysokoprzepustowych" oraz „Mikromacierze" prowadzonych na makrokierunku Bioinformatyka (Politechnika Poznańska oraz Uniwersytet Adama Mickiewicza).

Referencje

[B46] de Bruijn, N. A combinatorial problem. Proceedings of the Koninklijke Nedelandse Akademie van Wetenschappen*, 49* , 758–764, 1946.

[BFK+99a] J. Blazewicz, P. Formanowicz, M. Kasprzak, W.T. Markiewicz, and J. Weglarz. DNA sequencing with positive and negative errors. J. Comput. Biol., 6:113–123, 1999a.

[BFK+99b] Błażewicz J., Formanowicz P., Kasprzak M., Markiewicz W. T., Węglarz J., *Method of sequencing of nucleic acids*. Polish Patent Application P335786. 1999b.

[BGH+13] Burke EK, Gendreau M, Hyde M, Kendall G, Ochoa G, Ozcan E, Qu R. Hyper-heuristics: A survey of the state of the art. Journal of the Operational Research Society. 2013;**64**:1695-1724.

[BHK+99] Blazewicz J, Hertz A, Kobler D, de Werra D. On some properties of DNA graphs. *Discrete Applied Mathematics, 98* , 1–19, 1999.

[BK03] Blazewicz J, Kasprzak M. Complexity of DNA sequencing by hybridization. Theoretical Comput. Sci., 290:1459–1473, 2003.

[BMM+07] Burke EK, McCollum B, Meisels A, Petrovic S, Qu R. A graph-based hyperheuristic for educational timetabling problems. European Journal of Operational Research **176**:177-192. 2007.

[CRB10] Cano-Belmán J, Ríos-Mercado R, Bautista J. A scatter search based hyperheuristic for sequencing a mixed-model assembly line. Journal of Heuristics **16**:749-770. 2010.

[CKS01] Cowling P, Kendall G, Soubeiga E. A hyperheuristic approach for scheduling a sales summit. In: Selected Papers of the 3rd International Conference on the Practice and Theory of Automated Timetabling, PATAT 2000. Berlin: Springer. pp. 176-190. 2001.

[FKB+13] Frohmberg W, Kierzynka M, Blazewicz J, Gawron P, Wojciechowski P. G-DNA – a highly efficient multi- GPU/MPI tool for aligning nucleotide reads. Bull Pol Acad Sci:Tech. 61:989-992. 2013.

[GMA80] John Gallant, David Maier, and James Astorer. On finding minimal length superstrings. *Journal of Computer and System Sciences*, 20(1):50–58, 1980.

[GR10] Garrido P, Riff M. Dvrp: A hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic. Journal of Heuristics **16**:795-834. 2010.

[HGP03] Human Genome Project. Genomics and its impact on science and society: A 2003 primer. Technical report, U.S. Department of Energy, http://www.ornl.gov/sci/techresources/ HumanGenome/publicat/primer2001/index.shtml, 2003.

[IHGSC01] International Human Genome Sequencing Consortium. Initial sequencing and analysis of the human genome. *Nature* 409, p. 860–921, 2001.

[KG04] Krasnogor N, Gustafson S. A study on the use of "self-generation" in memetic algorithms. Natural Computing **3**(1):53-76. 2004.

[KP07] Keller RE, Poli R. Cost-benefit investigation of a genetic-programming hyperheuristic. In: Monmarche N, Talbi E-G, Collet P, Schoenauer M, Lutton E, editors. International Conference on Artificial Evolution. Berlin, Heidelberg: Springer-Verlag. pp. 13-24. 2007.

[LFK+88] Lysov Y, Florentiev V, Khorlin A, Khrapko K, Shik V, and Mirzabekov A. Determination of the nucleotide sequence of DNA using hybridization with oligonucleotides. A new method. Dokl Akad Nauk SSSR, 303:1508-1511. 1988.

[MG77] Maxam AM, Gilbert W. A new method for sequencing DNA. Proc. Natl. Acad. Sci. USA, 74:560–564, 1977.

[Pev89] P. Pevzner. l-tuple DNA sequencing: computer analysis. J Biomol Struct Dyn, 7:63-73. 1989

[PFU99] Preparata FP, Frieze AM, Upfal E. Optimal reconstruction of a sequence from its probes. J. Comput. Biol. 7, 361–368. 1999.

[Pil09] Pillay N. Evolving hyper-heuristics for the uncapacitated examination timetabling problem. In: Blazewicz J, Drozdowski M, Kendall G, McCollum B (eds). Multidisciplinary International Conference on Scheduling: Theory and Applications (MISTA'09). Dublin, Ireland. pp. 447-457. 2009.

[POH+14] Pappa GL, Ochoa G, Hyde MR, Freitas AA, Woodward J, Swan J. Genetic Programming and Evolvable Machines. Contrasting meta-learning and hyper-heuristic research: The role of evolutionary algorithms **15**(1):3-35. 2014.

[PR07] Pisinger D, Ropke S. A general heuristic for vehicle routing problems. Computers and Operations Research **34**(8):2403-2435. 2007.

[PU01] Preparata FP, Upfal E. *System and methods for sequencing by hybridization.* United States Patent Application US 2001/0004728. 21/07/2001. 2001.

[Run09] Runka A. Evolving an edge selection formula for ant colony optimization. In: Genetic and evolutionary computation conference (GECCO'09). New York: ACM; pp. 1075-1081. 2009.

[SAQ+12] Sabar NR, Ayob M, Qu R, Kendall G. A graph coloring constructive hyper-heuristic for examination timetabling problems. Applied Intelligence **37:**1-11. 2012.

[Sou88] Southern EM. United Kingdom Patent Application GB8810400, 1988.

[SNC77] Sanger F, Nickelen S, Coulson AR. DNA sequencing with chain-terminating inhibitors. Proc. Natl. Acad. Sci. USA, 74:560–564, 1977.

[TM12] Tabataba FS, Mousavi SR. A hyper-heuristic for the longest common subsequence problem. Computational Biology and Chemistry **36**:42-54. 2012.

[WC53] Watson JD, Crick FHC. Molecular structure of nucleic acids. Nature, 171(4356):737–738, 1953.

[WJH+81] Wallace RB, Johnson MJ, Hirose T, Miyake T, Kawashima EH, Itakura K. The use of synthetic oligonucleotides as hybridization probes. II. Hybridization of oligonucleotides of mixed sequence to rabbit beta-globin DNA. Nucleic Acids Res. 9, 879–894. 1981.

Aleksandra Śnieg

.....

(podpis)