STUDIA INFORMATICA 2010
Volume 31 Number 2A (89)

Dominika WIECZOREK, Bożena MAŁYSIAK-MROZEK, Dariusz MROZEK
Politechnika Śląska, Instytut Informatyki

JĘZYK ZAPYTAŃ DLA MOLEKULARNYCH STRUKTUR BIAŁKOWYCH*

Streszczenie. Reprezentacja białek w postaci struktury drugorzędowej dostarcza ważnych informacji dotyczących budowy białek i jest często wykorzystywana w procesie poszukiwania podobieństwa białek. Ponieważ istniejące komercyjne systemy zarządzania bazami danych nie udostępniają zintegrowanych sposobów zaawansowanego przetwarzania danych biologicznych na poziomie języka SQL, proces poszukiwania podobieństwa strukturalnego jest zwykle realizowany przez narzędzia zewnętrzne. W niniejszym artykule przedstawiono opracowany i zaimplementowany przez autorów język PSS-SQL, pozwalający na poszukiwanie w bazie danych białek o strukturze drugorzędowej podobnej do struktury wyspecyfikowanej przez użytkownika. W ten sposób otrzymujemy łatwy w zapisie, deklaratywny język do poszukiwania podobieństwa strukturalnego białek.

Słowa kluczowe: bioinformatyka, białka, struktura drugorzędowa, podobieństwo

QUERY LANGUAGE FOR PROTEIN MOLECULAR STRUCTURES

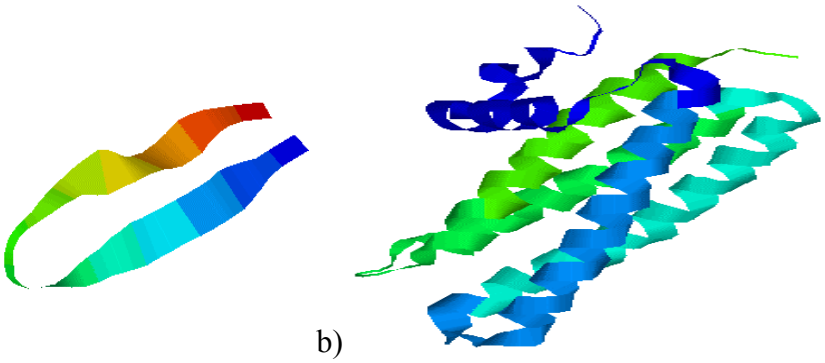
Summary. Secondary structure representation of proteins provides important information regarding protein general construction and shape. This representation is often used in protein similarity searching. Since existing commercial database management systems do not offer integrated exploration methods for biological data e.g. at the level of the SQL language, the structural similarity searching is usually performed by external tools. In the paper, we present our newly developed PSS-SQL language, which allows searching the database in order to identify proteins having secondary structure similar to the structure specified by the user in a PSS-SQL query. Therefore, we provide a simple and declarative language for protein structure similarity searching.

Keywords: bioinformatics, proteins, secondary structure, similarity

* Praca naukowa finansowana ze środków na naukę w latach 2008-2010 jako projekt badawczy

1. Wprowadzenie

Struktury drugorzędowe białek są cennym źródłem informacji o budowie tych niezwykle istotnych z punktu widzenia biologii komórkowej cząstek. Struktury drugorzędowe pozwalają bowiem ocenić ogólny kształt białka, ogólny sposób uformowania łańcucha aminokwasów, a także wskazać rodzaje lub elementy struktur drugorzędowych (SSE) występujących w jego budowie – czy są to struktury tylko jednego rodzaju, np. tylko α-helisy lub tylko β-kartki, czy struktury te są rodzajowo zróżnicowane. Ten sposób reprezentacji umożliwia również stwierdzenie, czy struktury danego typu są mocno posegregowane, czy występują naprzemiennie [1], [2]. Obserwacja trójwymiarowej budowy białka reprezentowanej na poziome struktur drugorzędowych ujawnia wzajemną organizację przestrzenną poszczególnych fragmentów białka i dostarcza informacji dotyczących formowania różnego rodzaju motywów strukturalnych (zwanych inaczej super-strukturami drugorzędowymi), np. popularna β-spinka (ang. *β-hairpin*) składa się z dwóch antyrównoległych β-nici (ang. *β-strand*) połączonych krótką pętlą (ang. *loop*, *turn* lub *coil*), równie często występującym motywem jest np. motyw β-α-β. Obserwacja białka w postaci struktury drugorzędowej pozwala również określić miejsca występowania domen funkcyjnych, czyli stabilnych konstrukcyjnie fragmentów białka, które mogą fałdować się niezależnie od siebie i pełnią zwykle określoną rolę w procesach komór-kowych [3], [4]. Na rysunku 1a przedstawiono fragment struktury formujący charakterystyczną β-spinkę, natomiast na rysunku 1b można obserwować struktury typu α-helisa w prze-strzennej budowie przykładowego białka pobranego z bazy Protein Data Bank (PDB) [5].



a) b)

Rys. 1. Przykłady struktur przestrzennych białek reprezentowanych na poziomie struktury drugo-rzędnowej: a) motyw β-spinka, b) struktury α-helikalne w białku PDB ID: 1X91 (crystal structure of mutant form A of a pectin methylesterase inhibitor from *Arabidopsis*)

Fig. 1. Sample protein spatial structures represented on the level of secondary structure: a) β-hairpin motif, b) α-helices in the structure of the protein PDB ID: 1X91 (crystal structure of mutant form A of a pectin methylesterase inhibitor from *Arabidopsis*)

Z tych powodów reprezentacja białek w postaci elementów struktury drugorzędowej stała się bardzo ważna dla analizy budowy i funkcji białek. Określenie rodzaju struktury

drugorzędowej, w skład której wchodzi dany aminokwas, jest m.in. często wykorzystywane w poszukiwaniu podobieństwa strukturalnego białek jako jeden z etapów całego procesu, m.in. w pracach [6-9]. Proces poszukiwania jest bowiem bardzo czasochłonny ze względu na fakt, że strukturę białka tworzą setki aminokwasów, a co za tym idzie – tysiące atomów połączonych wzajemnie wiązaniami kowalencyjnymi.

Poszukiwanie podobieństwa strukturalnego białek odbywa się poprzez porównanie struktury zadanego białka ze strukturami białek zgromadzonymi w bazie danych [10]. Proces ten jest zwykle realizowany przez narzędzia zewnętrzne. Dane opisujące struktury białek są natomiast przechowywane w systemach zarządzania bazami danych (SZBD), które świetnie spisują się w komercyjnych zastosowaniach, jednakże nie są przystosowane do przechowywania i przetwarzania danych biologicznych. Systemy te nie zapewniają natywnego wsparcia dla przetwarzania danych biologicznych m.in. na poziomie języka SQL, który jest podstawowym, deklaratywnym sposobem manipulacji danymi w systemach baz danych. Istnieje bardzo niewiele rozwiązań, które na poziomie bazy danych umożliwiają zaawansowane przetwarzanie danych biologicznych. Nie są to najczęściej rozwiązania komercyjne, a jedynie prototypy rozszerzeń wprowadzanych do istniejących systemów zarządzania bazami danych.

W niniejszym artykule przedstawiamy rozszerzenie języka SQL, umożliwiające zadawanie zapytań dotyczących struktury przestrzennej białek reprezentowanej przez elementy struktury drugorzędowej. Opracowany język o nazwie PSS-SQL (ang. *Protein Secondary Structure – Structured Query Languaage*) pozwala na poszukiwanie w bazie danych białek o strukturze drugorzędowej, podobnej do struktury wyspecyfikowanej przez użytkownika w zapytaniu PSS-SQL w postaci wzorca. Otrzymujemy dzięki temu zintegrowany z serwerem bazy danych deklaratywny sposób poszukiwania białek posiadających określony wzorzec strukturalny.

W kolejnych rozdziałach zebrano najważniejsze informacje dotyczące opracowanego języka PSS-SQL. Artykuł jest zorganizowany w następujący sposób: w rozdziale 2 przedstawiono istniejące języki umożliwiające poszukiwanie podobieństwa strukturalnego białek; w rozdziale 3 omówiono sposób reprezentacji, przechowywania i przygotowania danych przed przystąpieniem do formułowania zapytań PSS-SQL; w rozdziale 4 przedstawiono zmodyfikowany algorytm Smitha-Watermana używany do porównania struktur drugorzędowych białek w trakcie realizacji zapytań PSS-SQL; język PSS-SQL i sposób reprezentacji wzorca w procesie wyszukiwania zostały omówione w rozdziale 5; natomiast dyskusję na temat wydajności opracowanego języka przeprowadzono w rozdziale 6.

2. Istniejące rozwiązania

Przeszukiwanie dużych zbiorów danych przechowujących informacje o białkach, w celu wyszukania w bazie danych białek podobnych do zadanego wzorca, może być realizowane na wielu płaszczyznach. W szczególności bazując na ich strukturze drugorzędowej. Prace w tej dziedzinie są realizowane przez niewielu naukowców w różnych ośrodkach naukowo-badaw-czych. Powstają specjalne języki zapytań i rozszerzenia standardowego języka SQL, pozwalające na wyszukiwanie białek podobnych do zadanego, na poziomie sekwencji, struktury drugorzędowej czy trzeciorzędowej.

W pracy [11] autorzy rozszerzyli standardowy język SQL o elementy pozwalające na wyszukiwanie białek zawierających podane wzorcem fragmenty struktury drugorzędowej (podzielone na segmenty). Rozszerzenia zostały zaimplementowane zarówno w tworzonym przez nich systemie Periscope (silnik umożliwiający przetwarzanie zapytań dotyczących pierwszo-, drugo- i trzeciorzędowych struktur białkowych), jak również w komercyjnym systemie, jakim jest SZBD Oracle. Autorzy zaimplementowali również środowisko do optymalizacji skonstruowanych zapytań. W pracy założono, że struktura drugorzędowa reprezentowana jest w postaci następujących po sobie segmentów o różnych typach, np. hhhllleee itp. Informacje o segmentach przechowywane są w specjalnie utworzonej do tego celu tabeli. Wadą tego rozwiązania, z punktu widzenia użytkownika, jest bardzo złożony zapis, samo konstruowanie warunków filtrujących wymaga żmudnego pisania i w efekcie końcowym przyjmuje dość nieczytelną formę.

Prowadzone prace dotyczą również często rozszerzeń języka SQL w odniesieniu do wyszukiwania struktury pierwszorzędowej (sekwencji) białek. W pracy [12] autorzy rozszerzają system Periscope o kolejną część, zwaną Periscope/SQ. Periscope/SQ jest deklaratywnym narzędziem służącym do efektywnego przetwarzania zapytań dotyczących sekwencji białek. W tym celu wprowadzono nowe operatory, a rozszerzony język SQL nazwano PiQL. Zawiera on nowe typy danych oraz operatory algebraiczne, zgodne z utworzoną przez autorów algebrą PiQA. Dzięki tym rozszerzeniom w systemie istnieje możliwość konstruowania zapytań pozwalających na wyszukanie białka o zadanym wzorcem fragmencie sekwencji oraz spełniającym pewne kryteria dotyczące m.in. struktury drugorzędowej, np. zawierającej pętlę o długości 5. Wadą tego rozwiązania jest skomplikowana struktura, trudna do zapamiętania dla użytkownika, nie będącego informatykiem.

Podobne rozwiązania powstają także w obiektowych bazach danych. W pracy [13] autorzy rozszerzyli funkcjonalność systemu zarządzania obiektowymi bazami danych (SZOBD) o możliwość tworzenia zapytań dotyczących struktury białek, jak również przechowywania takich danych. W tym celu skonstruowali dwa komponenty: język zapytań Protein-QL oraz Protein-OODB – pozwalający na przechowywanie odpowiednich struktur danych i stano-

wiący sprzęg między Protein-QL a obiektową bazą danych. Autorzy utworzyli trzy nowe typy danych, służące do opisu odpowiednio pierwszo-, drugo- i trzeciorzędowej struktury białka. Jako bazę danych wybrano obiektową bazę Lambda-DB, w niej zaimplementowano język Protein-QL. Zapytania konstruowane w Protein-QL są kompilowane do OQL, który jest wykonywany już w bazie danych. Skonstruowane rozszerzenia pozwalają na wyszukanie dowolnej struktury białka, określonego warunkami (np. nazwą).

3. Sposób reprezentacji i przechowywania danych

Poszukiwanie białek o strukturze podobnej do zadanego wzorca poprzez formułowanie zapytań w języku PSS-SQL wymaga, aby dane o strukturach drugorzędowych białek były przechowywane w bazie danych w określonym formacie. W prezentowanym rozwiązaniu przyjęto, że struktury drugorzędowe białek będą reprezentowane przez sekwencje elementów struktury drugorzędowej (SSE). Każdy element struktury drugorzędowej będzie odpowiadał jednemu elementowi struktury pierwszorzędowej, a zatem pojedynczemu aminokwasowi. Na rysunku 2 zaprezentowano sekwencję aminokwasów białka o nazwie *6-phosphogluconolactonase* w organizmie *Escherichia coli* oraz odpowiadającą jej sekwencję elementów struktury drugorzędowej. Poszczególne symbole w sekwencji elementów struktury drugorzędowej mają następujące znaczenie: H odpowiada α-helisie, E odpowiada β-kartce, C odpowiada pętli (dla pętli stosuje się również oznaczenie L).

```
A7ZY23
6PGL_ECOHS
6-phosphogluconolactonase OS=Escherichia coli O9:H4 (strain HS) GN=pgl PE=3 SV=1

MKQTVYIASPESQQIHVWNLNHEGALTLTQVVDVPGQVQPMVVSPDKRYLYVGVRPEFRVLAYRIAPDDGALTFAAESAL
PGSPTHISTDHQGQFVFVGSYNAGNVSVTRLEDGLPVGVVDVVEGLDGCHSANISPDNRTLWVPALKQDRICLFTVSDDG
HLVAQDPAEVTTVEGAGPRHMVFHPNEQYAYCVNELNSSVDVWELKDPHGNIECVQTLDMMPENFSDTRWAADIHITPDG
RHLYACDRTASLITVF

CCCEEEEECCCCEEEEEEECCCCEEEEEEEEEEECCCCCEEEECCCEEEEEECCCCCEEEEEEEECCCCCCHHHHHHHCC
CCCCCCEEECCCCEEEEECCCCCCCEEEEEEECCCCCCEEEEEEECCCCCCCCCCCCCCCEEEECCHHHHHEEEEECCCCC
CEEECCCCCEEEECCCCCCCEEECCCCEEEEEECCCCCCEEEEEECCCCCCEEEEEECCCCCCCCCCCCCCEEEECCCCC
CEEEECCCCCCCEEEE
```

Rys. 2. Przykładowa sekwencja aminokwasów białka *6-phosphogluconolactonase* w organizmie *Escherichia coli* oraz odpowiadająca jej sekwencja elementów struktury drugorzędowej

Fig. 2. Sample amino acid sequence of the protein *6-phosphogluconolactonase* in the *Escherichia coli* with the corresponding sequence of secondary structure elements

Tak przedstawioną strukturę drugorzędową można w łatwy sposób przechowywać w bazie danych. Źródło danych o budowie drugorzędowej białek może być dowolne – dane mogą pochodzić bezpośrednio z bazy PDB, z systemów zajmujących się klasyfikacją struktur białkowych, np. SCOP [14] lub CATH [15], jak również mogą być wygenerowane przez

programy do predykcji struktur drugorzędowych białek na podstawie struktury pierwszorzędowej. Istotne jest jednak, aby były zapisane w postaci sekwencji symboli H, E, C/L.

W badaniach prowadzonych przez autorów dane były przechowywane w bazie danych *Proteins* w tabeli *ProteinTbl*, której strukturę wraz z przykładowymi danymi przedstawiono na rysunku 3.

id	protAC	protID	name	length	primary	secondary
3294	P01903	2DRA_HUMAN	HLA class II...	254	MAISGVPVLGFF...	CCCCCEEECCCE...
3295	Q30631	2DRA_MACMU	HLA class II...	254	MAESGVPVLGFF...	CCCCCEEECCCE...
3296	P11887	2ENR_CLOTY	2-enoate red...	30	MKNKSLFEVIKI...	CCCCCCEEEEEC...
3297	Q01284	2NPD_NEUCR	2-nitropropa...	378	MHFPGHSSKKEE...	CCCCCCCCCHHH...

Rys. 3. Przykładowa sekwencja aminokwasów białka *6-phosphogluconolactonase* w organizmie *Escherichia coli* oraz odpowiadająca jej sekwencja elementów struktury drugorzędowej

Fig. 3. Sample amino acid sequence of the protein *6-phosphogluconolactonase* in the *Escherichia coli* with corresponding sequence of secondary structure elements

Poszczególne kolumny tabeli *ProteinTbl* oznaczają:

id – unikalny identyfikator białka w tabeli

protAC – numer inwentarzowy białka *Accession Number*

protId – identyfikator białka w bazie SwissProt (pole *IDentification*)

name – nazwa białka

length – długość białka mierzona w [aa]

primary – struktura pierwszorzędowa białka

secondary – struktura drugorzędowa białka

Zanim nastąpi wyszukiwanie białek podobnych do zadanego wzorca dane dotyczące struktury drugorzędowej białka muszą zostać poindeksowane. Indeks jest tworzony poprzez wykonanie procedury *usp_indexSS*, jak przedstawiono to w przykładzie:

```
EXEC dbo.usp_indexSS
@columnName = 'secondary',
@indexName = 'TIDX_Secondary';
```

Parametr *@columnName* wskazuje, która kolumna zawiera indeksowaną sekwencję SSE. Wykonanie procedury powoduje w pierwszym kroku utworzenie w bazie danych dodatkowej tabeli segmentów o nazwie określonej parametrem *@indexName*, w której znajdują się wyekstrahowane z sekwencji struktury drugorzędowej białka fragmenty struktury danego typu wraz z informacją o ich długości. Tabela segmentów jest wykorzystywana do przyspieszenia procesu wyszukiwania białek. Odbywa się to poprzez wstępne odfiltrowanie białek, których struktura nie wykazuje podobieństwa do zadanego wzorca. Filtrowanie to polega na tym, że na podstawie wzorca zdefiniowanego przez użytkownika wyciągane są najbardziej charakterystyczne cechy szukanej sekwencji SSE i dzięki wyszukaniu ich w tabeli segmentów możliwe jest ograniczenie liczby białek poddanych następnie dopasowaniu, które stanowi kolejną fazę procesu poszukiwania. Strukturę tabeli segmentów wraz

z przykładowymi danymi przedstawiono na rysunku. 4. Dane w powstałej tabeli są poindeksowane indeksem typu B-Tree.

id	protID	type	startPos	length
67	3	C	0	3
68	3	H	3	23
69	3	C	26	8
70	3	H	34	12
71	3	C	46	3
72	3	E	49	3

Rys. 4. Fragment tabeli segmentów
Fig. 4. Part of the segment table

Metadane opisujące powiązanie tabeli segmentów z konkretną kolumną opisującą strukturę drugorzędową zostają zapisane w tabeli mapującej *MappingTbl*, której postać zaprezentowano na rysunku 5.

id	columnName	indexName	maxLength
3	secondary	TIDX_Secondary	290

Rys. 5. Fragment tabeli mapującej
Fig. 5. Part of the mapping table

4. Zmodyfikowany algorytm Smitha-Watermana

W procesie wyszukiwania w bazie danych białek o strukturze podobnej do zadanego wzorca w postaci sekwencji elementów struktur drugorzędowych wykorzystywany jest algorytm Smitha-Watermana [16]. Autorzy zmodyfikowali pierwotną wersję tego algorytmu, przewidzianą do dopasowania sekwencji nukleotydów i sekwencji aminokwasów, przystosowując go do dopasowania sekwencji SSE. Ponadto zmodyfikowana wersja algorytmu zwraca nie jedno optymalne rozwiązanie, ale wiele takich rozwiązań ze względu na przybliżony charakter wzorca. W niniejszym rozdziale omówiono działanie zmodyfikowanego algorytmu Smitha-Watermana.

Dane są dwa białka *A* i *B*. Struktury pierwszorzędowe (sekwencje aminokwasów) białek *A* i *B* można przedstawić w postaci:

$P^A = p_1^A p_2^A ... p_n^A$ oraz $P^B = p_1^B p_2^B ... p_m^B$,

gdzie: *n* jest długością białka *A* mierzoną w aminokwasach [aa], *m* jest długością białka *B*, $p_i \in P$, *P* jest zbiorem 20 znanych typów aminokwasów.

Struktury drugorzędowe białek *A* i *B* można przedstawić w postaci:

$S^A = s_1^A s_2^A ... s_n^A$ oraz $S^B = s_1^B s_2^B ... s_m^B$,

gdzie: $s_i \in S$ jest elementem struktury drugorzędowej (SSE) odpowiadającym *i*-temu aminokwasowi p_i, $S = \{H, E, C, ?\}$ jest zbiorem 3 typów struktur drugorzędowych:

- *H* odpowiada α-helisie,
- *E* odpowiada β-nici,
- *C* odpowiada tzw. pętli lub zakrętowi (ang. *loop* lub *turn*),
- ? odpowiada dowolnemu z wyżej wymienionych elementów.

W procesie dopasowania budowana jest macierz dopasowania *D* według następujących reguł:

dla $0 \le i \le n$ oraz $0 \le j \le m$:

$$D_{i,0} = D_{0,j} = 0, \tag{1}$$

$$D_{i,j}^{(1)} = D_{i-1,j-1} + \delta(s_i^A, s_j^B), \tag{2}$$

$$D_{i,j}^{(2)} = \max_{1 \le k \le n} \{D_{i-k,j} - \omega_k\}, \tag{3}$$

$$D_{i,j}^{(3)} = \max_{1 \le l \le m} \{D_{i,j-l} - \omega_l\}, \tag{4}$$

$$D_{i,j}^{(4)} = 0, \tag{5}$$

$$D_{i,j} = \max_{v=1..4} \{D_{i,j}^{(v)}\}, \tag{6}$$

gdzie: $\delta(s_i^A, s_j^B)$ jest nagrodą δ^+ w przypadku dopasowania dwóch elementów struktury drugorzędowej białek *A* i *B* lub karą za niedopasowanie δ^- w przypadku braku dopasowania:

$$\delta(s_i^A, s_j^B) = \begin{cases} 1 & gdy \quad s_i^A = s_j^B \\ -1 & gdy \quad s_i^A \neq s_j^B \end{cases}, \tag{7}$$

ω_k jest karą za wprowadzenie przerwy o długości *k*:

$$\omega_k = \omega_O + k \times \omega_E, \tag{8}$$

gdzie: $\omega_O = 3$ jest karą za otwarcie przerwy, $\omega_E = 0.5$ jest karą za rozszerzenie przerwy.

Na rysunku 6 przedstawiono system punktacji (nagrody/kary) dla poszczególnych par elementów struktury drugorzędowej.

	H	E	C	?
H	1	-1	-1	1
E	-1	1	-1	1
C	-1	-1	1	1
?	1	1	1	1

Rys. 6. System punktacji dla poszczególnych par elementów struktury drugorzędowej
Fig. 6. Scoring system for particular pairs of secondary structure elements

Obliczenie dopasowania polega na znalezieniu maksymalnej wartości w macierzy dopasowania *D* i powrocie *po śladach*, przechodząc przez komórki, które umożliwiły wyprowadzenie wartości danej komórki, aż do napotkania wartości 0. W procesie poszukiwania zadanego wzorca w strukturze drugorzędowej białka z bazy danych może

wystąpić więcej niż jedno dopasowanie. Zmodyfikowana wersja algorytmu zwraca wiele takich rozwiązań poprzez znajdowanie kolejnych maksimów w macierzy *D*, kierując się dodatkowym wewnętrznym parametrem – tzw. minimalną wartością końca ścieżki *MPE* (ang. *Minimum Path End*). Zatrzymanie odnajdywania kolejnych dopasowań następuje, gdy kolejne maksimum z macierzy dopasowania jest mniejsze od wartości parametru *MPE*. Wartość *MPE* jest uzależniona od zadanego w zapytaniu wzorca zgodnie z wyrażeniem:

$$MPE = (MPL \times \delta^{+}) + (NoIS \times \delta^{-}),$$

gdzie: *MPL* to minimalna długość wzorca (ang. *Minimum Pattern Length*), *NoIS* to liczba nieprecyzyjnie zdefiniowanych segmentów wzorca (ang. *Number of Imprecise Segments*), tzn. takich, w których minimalna długość bloku jest różna od maksymalnej długości bloku.

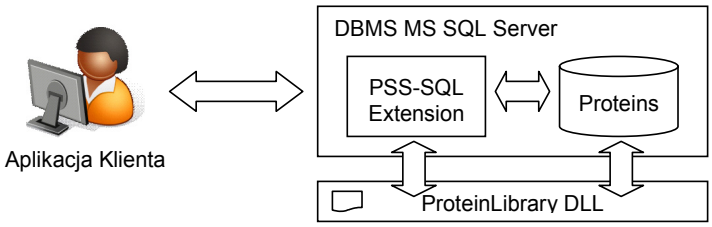
Na przykład, dla wzorca strukturalnego *h(10;20),e(1;10),c(5),e(5;20)* składającego się z α-helisy o długości 10 do 20 elementów, β-nici o długości 1 do 10 elementów, pętli o długości 5 elementów oraz β-nici o długości 5 do 20 elementów, parametr *MPL=21* (10 elementów typu *h*, 1 element typu *e*, 5 elementów typu *c* i 5 elementów typu *e*), parametr *NoIS=3* (pierwszy, drugi i czwarty segment), a zatem wartość *MPE=18*.

5. Rozszerzenia języka SQL o możliwość wyszukiwania białek o zadanej strukturze

Efektem prac autorów nad prostym i wygodnym sposobem wyszukiwania białek o zadanej strukturze jest opracowany i zaimplementowany deklaratywny język zapytań PSS-SQL (ang. *Protein Secondary Structure – Structured Query Languaage*). PSS-SQL rozszerza standard języka SQL o dodatkowe funkcje umożliwiające poszukiwanie białek o strukturze drugorzędowej identycznej lub podobnej do zadanej. Rozszerzenie PSS-SQL udostępnia dwie funkcje *containSequence* i *sequencePosition*, które umożliwiają prowadzenie poszukiwań. PSS-SQL, obejmuje także szereg procedur i funkcji pomocniczych, wykorzystywanych niejawnie, które m.in. ekstrahują segmenty struktur poszczególnych typów, tworzą tabele pomocnicze, indeksują dane o strukturach drugorzędowych, operują na tych strukturach, pozwalają na dopasowanie struktury z bazy danych do wzorca i wiele innych czynności pośrednich.

Rozszerzenie PSS-SQL zostało zaimplementowane w języku C# w środowisku .NET dla SZBD Microsoft SQL Server 2005. Wszystkie funkcje i procedury zostały zebrane w postaci biblioteki DLL o nazwie *ProteinLibrary*, umieszczanej na komputerze użytkownika podczas procesu instalacji. Po zainstalowaniu rozszerzenia PSS-SQL w bazie danych przechowującej dane o budowie białek zostają zarejestrowane procedury i funkcje operujące na poziomie natywnego w Microsoft SQL Server języka Transact-SQL. Na rysunku 7 przedstawiono

ogólną architekturę oraz przepływ informacji w systemie z zainstalowanym rozszerzeniem PSS-SQL.



Rys. 7. Ogólna architektura systemu z rozszerzeniem PSS-SQL
Fig. 7. General architecture of the system with the PSS-SQL extension

5.1. Reprezentacja wzorca w procesie wyszukiwania

Standardowy język zapytań SQL został wzbogacony o możliwość definiowania postaci szukanego wzorca sekwencji elementów struktury drugorzędowej białka, będącej jednym z parametrów zaimplementowanych funkcji *containSequence* i *sequencePosition*. Projektując dodatkowe funkcjonalności języka należało pamiętać, by wprowadzane rozszerzenia umożliwiały użytkownikowi formułowanie jak największej liczby typów zapytań o różnym stopniu złożoności oraz by forma rozszerzeń była jak najprostsza i nie nastręczała dodatkowych trudności z zapisem. Z tego powodu w celu konstruowania postaci wzorca sekwencji SSE białka zdefiniowano odpowiednią gramatykę, pozwalającą na generowanie tego typu sekwencji.

Sekwencja elementów struktury drugorzędowej białka jest reprezentowana przez następujące po sobie bloki segmentów. Każdy z segmentów określony jest przez typ oraz jego długość. Długość segmentu może zostać przedstawiona w postaci przedziału lub konkretnego rozmiaru. Można także zdefiniować segmenty, dla których typ nie jest istotny (wstawia się wtedy symbol '?') oraz takie, dla których wartość końcowa przedziału nie jest określona (symbol '*'). Przykładowa gramatyka zapisana w notacji Chomskiego, definiująca sekwencję elementów struktury drugorzędowej białka, może mieć następującą postać:

$G_{pss} = <V_{pss}, \Sigma_{pss}, P_{pss}, \sigma_{pss}>$,

gdzie odpowiednio symbole oznaczają:

V_{pss} – zbiór symboli terminalnych,

Σ_{pss} – zbiór symboli nieterminalnych (pomocniczych),

P_{pss} – zbiór reguł, określających jak wyprowadza się słowa,

σ_{pss} – symbol startowy (głowa).

```
Vpss = {c, h, e, ?, *, N}
Σpss = {<sekwencja>, <bloki_segmentów>, <segment>, <typ>, <początek>, <koniec>,
        <długość>, <liczba_naturalna_dodatnia_i_zero>, <nieokreślony>}

Ppss = {<sekwencja> ::= <bloki_segmentów>
    <bloki_segmentów> ::= <segment> | <segment>, <bloki_segmentów>
```

```
<segment> ::= <typ> (<początek>; <koniec>) | <typ> (<długość>)
<początek> ::= <liczba_naturalna_dodatnia_i_zero>
<koniec> ::= <liczba_naturalna_dodatnia_i_zero> | <nieokreślony>
<długość> ::= <liczba_naturalna_dodatnia_i_zero>
<typ> ::= c | h | e | ?
<liczba_naturalna_dodatnia_i_zero> ::= N+ | 0
<nieokreślony> ::= * }
```

σ_{pss} = <sekwencja>

Założenie:

<początek> <= <koniec>

Przedstawione poniżej słowa są poprawne względem zdefiniowanej gramatyki G_{pss}:

e(10; 15)
c(25), h(25; 40), ?(10)
*h(5; *), c(25), e(35, 60)*.

5.2. Funkcja *containSequence*

Funkcja *containSequence* pozwala sprawdzić, czy wskazane białko lub zbiór białek z bazy danych zawiera podany w zapytaniu wzorzec struktury w postaci sekwencji elementów struktur drugorzędowych. Funkcja zwraca wartość logiczną 1, jeśli białko z bazy danych zawiera szukany wzorzec, lub 0, jeśli białko nie zawiera w swojej strukturze zadanego wzorca.

Nagłówek funkcji *containSequence* jest następujący:

```
FUNCTION containSequence
(
@proteinId int,
@columnSSeq text,
@pattern varchar(4000)
)
RETURNS bit
```

Funkcja *containSequence* przyjmuje trzy parametry wejściowe:
@proteinId – unikalny identyfikator białka z tabeli zawierającej sekwencje SSE (w przypadku tabeli *ProteinTbl* jest to kolumna *id*),
@columnSSeq – nazwa kolumny zawierającej sekwencje SSE (np. *secondary*),
@pattern – wzorzec wyszukiwanego fragmentu struktury, składający się z bloków segmentów, np. *e(1;5), c(10;15),?(2,*)*.

Funkcja może być wykorzystana zarówno w klauzuli SELECT polecenia SELECT języka PSS-SQL, jak i w klauzuli WHERE.

Zastosowanie funkcji w klauzuli SELECT pozwala zwrócić informację, czy dane białko lub grupa białek zawiera zadany wzorzec. Poniżej przedstawiono przykład użycia funkcji podczas weryfikacji występowania wzorca *c(10;20),e(7;20), c(1;20)* w strukturze wskazanego białka *Q9FHY1*.

```
SELECT id, protID, protAC, name,
containSequence(id,'secondary','c(10;20),e(7;20),c(1;20)') AS containSeq
FROM ProteinTbl
WHERE protAC='Q9FHY1'
```

Wynik zapytania został przedstawiony na rys. 8.

id	protID	protAC	name	containSeq
964	ABIL4_ARATH	Q9FHY1	Protein ABIL4 OS=Arabidopsis thaliana	0

Rys. 8. Wynik wyszukiwania dla wskazanego białka *Q9FHY1*
Fig. 8. Result of the searching process for the protein *Q9FHY1*

Kolejne zapytanie przedstawia przykład zastosowania funkcji *containSequence* dla białek z gatunku *Arabidopsis thaliana*.

```
SELECT id, protID, protAC, name,
containSequence(id,'secondary','c(10;20),e(7;20),c(1;20)') AS containSeq
FROM ProteinTbl
WHERE name like '%Arabidopsis thaliana%'
```

Wyniki zapytania zostały przedstawione na rys. 9.

id	protID	protAC	name	containSeq
175	A494_ARATH	P43295	Probable cysteine proteinase A494 OS=	1
244	A9_ARATH	Q00762	Tapetum-specific protein A9 OS=Arabid	0
443	AAH_ARATH	O49434	Allantoate deiminase, chloroplastic O	1
522	AASS_ARATH	Q9SMZ4	Alpha-aminoadipic semialdehyde syntha	1
553	AAT1_ARATH	P46643	Aspartate aminotransferase, mitochond	1
560	AAT2_ARATH	P46645	Aspartate aminotransferase, cytoplasm	1
...				

Rys. 9. Wyniki zapytania dla białek z gatunku *Arabidopsis thaliana*
Fig. 9. Result of the searching process for proteins from the *Arabidopsis thaliana*

Zastosowanie funkcji *containSequence* w klauzuli WHERE pozwala zwrócić białka, które zawierają zadany wzorzec. Poniżej przedstawiono przykład użycia funkcji podczas poszukiwania białek funkcjonujących w organizmie *Escherichia coli* i zawierających wzorzec strukturalny *h(5;15),c(3),?(6),c(1;*)*.

```
SELECT id, protID, protAC, name, primary, secondary
FROM ProteinTbl
WHERE containSequence(id, 'secondary','h(5;15),c(3),?(6),c(1;*)')=1
and name like '%Escherichia coli%'
```

Wyniki zapytania zostały przedstawione na rysunku 10.

id	protID	protAC	name	primary	secondary
1294	ACCA_ECO24	A7ZHS5	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHCCH...
1295	ACCA_ECO57	P0ABD6	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHCCH...
1296	ACCA_ECOHS	A7ZWD1	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHCCH...
1297	ACCA_ECOK1	A1A7M9	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHCCH...
1298	ACCA_ECOL5	Q0TLE8	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHCCH...
1299	ACCA_ECOL6	Q8FL03	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHCCH...
1300	ACCA_ECOLI	P0ABD5	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHHHH...
1301	ACCA_ECOUT	Q1RG04	Acetyl-coenzyme A ca...	MSLNFLDFEQPIAELEAKID...	CCCCCCCCCHHHHHHHHHHHHHHH...

Rys. 10. Wyniki poszukiwania białek funkcjonujących w organizmie *Escherichia coli* i zawierających zadany wzorzec strukturalny *h(5;15),c(3),?(6),c(1;*)*
Fig. 10. Result of the searching process for proteins from the *Escherichia coli* having the given structural pattern *h(5;15),c(3),?(6),c(1;*)*

5.3. Funkcja *sequencePosition*

Funkcja *sequencePosition* pozwala zlokalizować podany wzorzec w strukturze drugorzędowej białka lub grupy białek z bazy danych. Lokalizacja wzorca w strukturze białka z bazy danych jest prowadzona poprzez dopasowanie z użyciem zmodyfikowanego algorytmu Smitha-Watermana, przedstawionego w rozdziale 4.

Nagłówek funkcji *sequencePosition* jest następujący:

```
FUNCTION sequencePosition
(
@columnSSeq text,
@pattern varchar(4000),
@condition varchar(4000)
)
RETURNS @resultTable table
(
proteinId int,
startPos int,
endPos int,
length int,
gapsCount int,
sequence text
)
```

Funkcja *sequencePosition* przyjmuje trzy parametry wejściowe:
@columnSSeq – nazwa kolumny zawierającej sekwencje SSE, np. *secondary*,
@pattern – wzorzec wyszukiwanego fragmentu struktury, składający się z bloków segmentów, np. *e(1;5), c(10;15),?(2,*)*,
@condition – opcjonalny, prosty lub złożony warunek filtrujący, pozwalający ograniczyć listę białek z bazy danych, które zostaną przetworzone w celu zlokalizowania zadanego wzorca, np. *name LIKE '%phosphogluconolactonase%'*.

W wyniku działania funkcji *sequencePosition* otrzymujemy tabelę, której kolumny zawierają informacje o lokalizacji zadanego wzorca w strukturze kolejnego białka z bazy danych:
proteinId – unikalny identyfikator białka z bazy danych zawierający poszukiwany wzorzec; dzięki temu identyfikatorowi można złączyć tabelę wynikową funkcji z danymi z innych tabel,
startPos – pozycja, od której rozpoczyna się zadany wzorzec strukturalny w białku znalezionym w bazie danych,
endPos – pozycja, na której kończy się zadany wzorzec strukturalny w białku znalezionym w bazie danych,
length – długość segmentu dopasowanego do zadanego w zapytaniu wzorca strukturalnego,
gapsCount – liczba ewentualnych przerw w dopasowaniu,
sequence – sekwencja SSE dopasowana do zadanego w zapytaniu wzorca strukturalnego.

Funkcja *sequencePosition* jest wykorzystywana w klauzuli FROM polecenia SELECT języka PSS-SQL, gdzie jest traktowana jako jedna z tabel źródłowych, potrzebnych do realizacji zapytania. Poniżej przedstawiono przykład użycia funkcji podczas lokalizacji w strukturach białek znajdujących się w bazie danych wzorca strukturalnego składającego się z α-helisy o długości od 5 do 20 elementów, opcjonalnej pętli oraz dwóch β-nici oddzielonych opcjonalną pętlą – wzorzec *h(5;20),c(0;*),e(1;*),c(0;*),e(1;*)*. Wzorzec jest lokalizowany tylko w białkach o długości przekraczającej 100 aa, których struktura została przewidziana (warunek *PE=4*).

```
SELECT p.id, p.name, s.startPos, s.endPos, s.length, s.gapsCount,
s.sequence, p.secondary
FROM ProteinTbl p JOIN
dbo.sequencePosition('secondary','h(5;20),c(0;*),e(1;*),c(0;*),e(1;*)','')   AS
s ON p.id = s.proteinId
WHERE p.name LIKE '%PE=4%' AND p.length > 100
```

Wyniki zapytania zostały przedstawione na rysunku 11.

id	name	startPos	endPos	length	gapsCount	sequence	secondary
3298	2-nitropropane ...	330	350	20	0	hhhhhccccccccceeeeee	ccchhhhhheeeee...
3298	2-nitropropane ...	244	309	65	0	hhhhhhhhhhhhhhhhhhhhccccccccc	ccchhhhhheeeee...
3298	2-nitropropane ...	244	302	58	0	hhhhhhhhhhhhhhhhhhhhcccccccc	ccchhhhhheeeee...
3298	2-nitropropane ...	110	148	38	0	hhhhhhhhhhccccccccccceeeeccc	ccchhhhhheeeee...
3298	2-nitropropane ...	171	213	42	0	hhhhhhhhhhhhhcceeeeeeccccccc	ccchhhhhheeeee...
3298	2-nitropropane ...	214	238	24	0	hhhhhhhhhhcccccccceeee	ccchhhhhheeeee...
3298	2-nitropropane ...	110	136	26	0	hhhhhhhhhhcccccccccccceee	ccchhhhhheeeee...
3298	2-nitropropane ...	153	171	18	0	hhhhhhhhcceeeeeee	ccchhhhhheeeee...
3298	2-nitropropane ...	4	16	12	0	hhhhhheeeee	ccchhhhhheeeee...
3918	Acetoin utiliza...	115	146	31	0	hhhhhhhhhhhhhcccccccccceeeeee	ccchhhhhheeeee...
3918	Acetoin utiliza...	60	86	26	0	hhhhhhhhhhcccccccccceeeee	ccchhhhhheeeee...
3918	Acetoin utiliza...	149	187	38	0	hhhhhhhhhhheeeeeeeeeeccccce	ccchhhhhheeeee...
3918	Acetoin utiliza...	149	172	23	0	hhhhhhhhhhheeeeeeeeee	ccchhhhhheeeee...
3918	Acetoin utiliza...	90	111	21	0	hhhhhhhhhhcccccceeee	ccchhhhhheeeee...
3918	Acetoin utiliza...	194	207	13	0	hhhhhcccceeee	ccchhhhhheeeee...
3918	Acetoin utiliza...	4	16	12	0	hhhhheeeeee	ccchhhhhheeeee...
...							

Rys. 11. Wyniki poszukiwania białek zawierających zadany wzorzec strukturalny *h(5;20),c(0;*),e(1;*),c(0;*),e(1;*)*

Fig. 11. Result of the searching process for the given structural pattern *h(5;20),c(0;*),e(1;*),c(0;*),e(1;*)*

Warto zwrócić uwagę, że może istnieć wiele sposobów dopasowania zadanego wzorca do struktury białka z bazy danych. Zmodyfikowany algorytm Smitha-Watermana zwraca wiele możliwych dopasowań, kierując się wartością wewnętrznego parametru *MPE*. Stąd też w tabeli wyników pokazanej na rysunku 11 to samo białko może pojawić się wielokrotnie z różnymi parametrami dotyczącymi lokalizacji wzorca.

Warunki filtrujące zbiór wierszy mogą być definiowane zarówno w klauzuli WHERE polecenia SELECT, jak i podane jako parametr *@condition* funkcji *sequencePosition*. Z punktu widzenia wydajności zapytań lepiej jest jednak przekazywać je w postaci parametru *@condition* bezpośrednio podczas wywołania funkcji. Powoduje to przeprowadzenie procesu filtrowania danych przed konstruowaniem tabeli wynikowej i wykonaniem algorytmu Smitha-Watermana. Przykład zapytania z warunkami filtrującymi podanymi podczas wywołania funkcji przedstawiono poniżej.

```
SELECT p.id, p.name, s.startPos, s.endPos, s.length, s.gapsCount,
s.sequence, p.secondary
FROM ProteinTbl p JOIN
dbo.sequencePosition('secondary','h(5;20),c(0;*),e(1;*),c(0;*),e(1;*)',
'p.name like ''%PE=4%'' AND p.length > 100') AS s
ON p.id = s.proteinId
```

6. Wydajność zapytań PSS-SQL

Podczas prac projektowych i implementacyjnych nad powstaniem języka PSS-SQL przeprowadzono badania, których celem była weryfikacja skuteczności dopasowania zadanego wzorca do struktur białek przechowywanych w bazie danych, a także określenie wydajności zaproponowanego rozwiązania.

Skuteczność dopasowania została zweryfikowana manualnie poprzez porównanie wyników zwracanych przez funkcje *containSequence* oraz *sequencePosition* z sekwencjami SSE zapisanymi w kolumnie *secondary* tabeli *ProteinTbl* oraz układem segmentów zapisanym w tabeli segmentów. Weryfikacji dokonano dla ponad 100 różnych wzorców zapytań, o różnym stopniu złożoności, składających się z różnej liczby bloków segmentów, określonych w sposób dokładny lub przybliżony, zawierających elementy SSE dokładnie zdefiniowane lub dowolne.

Badania wydajności zapytań PSS-SQL zostały przeprowadzone na komputerze stacjonarnym klasy PC, z procesorem Intel® 3.2 GHz Core Duo oraz pamięcią RAM 2GB. Baza danych *Proteins*, którą posłużono się podczas badań, zawierała dane o strukturach pierwszorzędowych i drugorzędowych 6230 białek. Dane o strukturach pierwszorzędowych białek zostały pobrane z bazy SwissProt [17], natomiast struktury drugorzędowe tych białek zostały wygenerowane w procesie predykcji przy użyciu programu Predator [18].

Wykonanie zapytania z funkcją *sequencePosition*, a zatem pozwalającego na lokalizację wzorca w strukturze białek z bazy danych, może zająć od ułamków sekund do kilku minut. Czas ten zależy ściśle od zadanego wzorca. Na rysunku 12a zaprezentowano zależność czasu wykonania zapytania od liczby segmentów wzorca struktury drugorzędowej białka podanego w zapytaniu użytkownika. Przedstawiono wyniki dla zapytań zawierających przykładowe wzorce złożone z pojedynczych segmentów, które następnie stopniowo rozbudowywano do ostatecznej postaci:

- SSE1: *h(38),c(3;10),e(25;30),c(3;10),h(1;10),c(1;5),e(5;10)*
- SSE2: *e(4;20),c(3;10),e(4;20),c(3;10),e(15),c(3;10),e(1;10)*
- SSE3: *h(30;40),c(1;5),?(50;60),c(5;10),h(29),c(1;5),h(20;25)*
- SSE4: *h(10;20),c(1;10),h(243),c(1;10),h(5;10),c(1;10),h(10;15)*

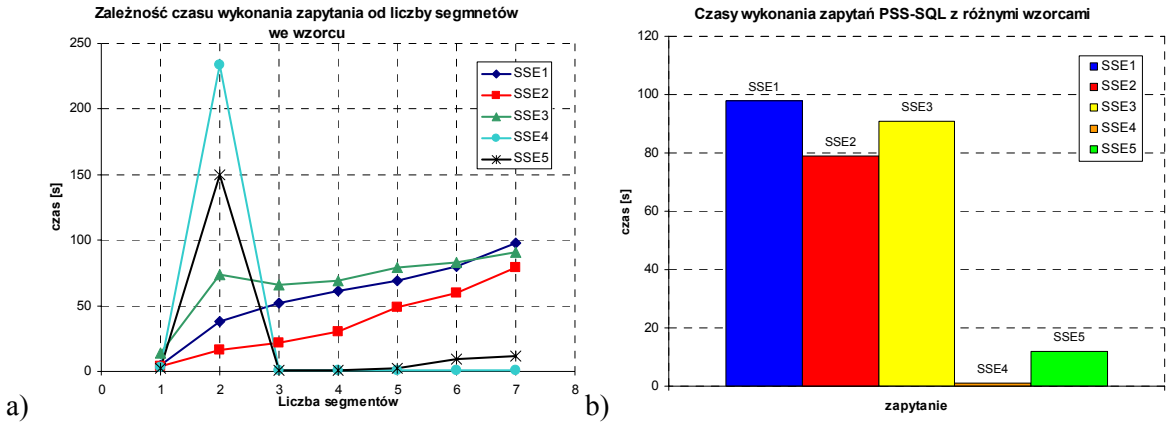
- SSE5: *e(1;10),c(1;5),e(27),h(1;10),e(1;10),c(1;10),e(5;20)*

Wzorzec SSE1 reprezentuje tutaj struktury zawierające naprzemiennie ułożone regiony typu α-helisa i β-nić oddzielone pętlami. Wzorzec SSE2 reprezentuje strukturę składającą się tylko z regionów typu β-nić oddzielonych pętlami. Natomiast wzorzec SSE3 zawiera blok elementów nieokreślonego typu. Wzorce SSE4 oraz SSE5 zawierają jeden unikalny, charakterystyczny region, odpowiednio *h(243)* i *e(27)*. Na rysunku 12a można zaobserwować, że złożoność wzorca oraz jego długość pełnią rolę drugoplanową, jeśli chodzi o wpływ tych parametrów na czas wykonania zapytania. Elementem istotnie wpływającym jest natomiast stopień unikalności wzorca. Zwiększenie unikalności wyszukiwanego wzorca powoduje odfiltrowanie większej liczby białek na podstawie informacji z tabeli segmentów, a co za tym idzie, zmniejszenie liczby białek podlegających dopasowaniu oraz skrócenie czasu realizacji zapytania. Widać to wyraźnie na rysunku 12b na przykładzie wzorców SSE4 i SSE5, zawierających dokładnie zdefiniowane, rzadko występujące regiony *h(243)* i *e(27)*. Dla wzorców uniwersalnych, do których można znaleźć wiele pasujących białek i/lub wielokrotne dopasowania, stopień skomplikowania wzorca oraz jego długość zaczynają mieć większy wpływ na czas wykonania zapytania PSS-SQL. Obserwuje się zatem wzrost czasów wykonania tego typu zapytań wraz ze wzrostem liczby segmentów i wzrostem ogólnej długości wzorca (rys. 12a). Dzieje się tak na skutek wzrostu złożoności procesu filtrowania białek na podstawie zapisów tabeli segmentów. Zwiększenie długości wzorca wpływa natomiast na wydłużenie procesu dopasowania przy użyciu zmodyfikowanej wersji algorytmu Smitha-Watermana. Nie zaobserwowano bezpośredniego wpływu typu struktury drugorzędowej na przebieg procesu dopasowania i czas wykonania zapytań.

Bezpośredni wpływ na czas wykonania zapytania mają również dodatkowe warunki filtrujące zdefiniowane w klauzuli WHERE zapytania lub przekazane jako parametr funkcji (tylko w przypadku *sequencePosition*). W przypadku funkcji *containSequence* dodatkowe warunki filtrujące przyspieszają zwykle proces pobierania właściwych danych. Warunki te podaje się tylko w klauzuli WHERE.

W przypadku funkcji *sequencePosition* dodatkowe warunki można precyzować zarówno w klauzuli WHERE, jak i poprzez parametr funkcji. Jednak przekazanie ich poprzez parametr funkcji daje znacznie większe przyspieszenie niż w przypadku klauzuli WHERE, gdzie przyspieszenie w niektórych przypadkach może być nawet nieodczuwalne. Dzieje się tak na skutek tego, że warunki klauzuli WHERE są nakładane na tabelę wynikową funkcji *sequencePosition*, po jej skonstruowaniu. Warunki przekazywane przez parametr są natomiast uwzględniane przed konstruowaniem tabeli wynikowej. Na rysunku 13 przedstawiono czasy wykonania zapytań PSS-SQL poszukujących wzorca strukturalnego SSE1: *h(38),c(3;10),e(25;30),c(3;10),h(1;10),c(1;5),e(5;10)*, wraz z dodatkowymi predykatami filtrującymi zdefiniowanymi jako parametr funkcji (BUILT-IN) lub w klauzuli WHERE:

- predykat P1: *p.name like "%Homo sapiens%"*
- predykat P2: *p.name like "%Homo sapiens%PE=1%"*
- predykat P3: *p.name like "%Homo sapiens%PE=1%SV=4%"*
- predykat P4: *p.primary like "%NHSAAYRVDQGVLN%"*

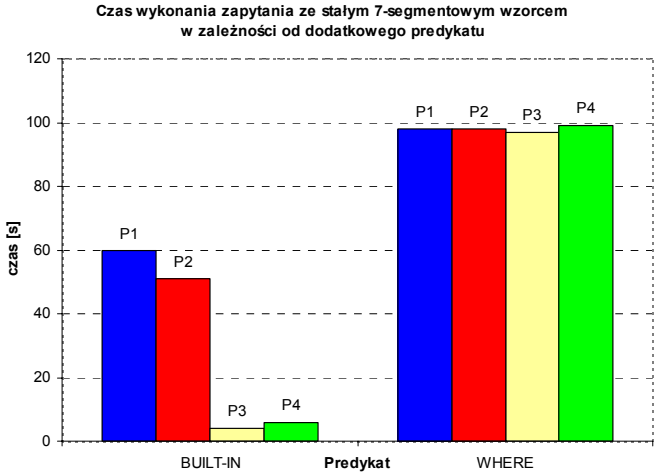


Rys. 12. Czasy wykonania zapytań w języku PSS-SQL: a) zależność czasu wykonania od liczby segmentów we wzorcu, b) czasy wykonania zapytań dla 7-segmentowych wzorców

Fig. 12. Execution times for PSS-SQL queries: a) dependency between the time and the number of segments in query pattern, b) execution times for queries with patterns containing 7 different segments

Dodatkowy predykat P1 powoduje, że porównanie wzorca strukturalnego będzie prowadzone tylko dla białek pełniących swe funkcje w organizmach z gatunku *Homo sapiens*. W predykacie P2 dodano warunek, że poszukiwanie nastąpi tylko wśród białek, dla których istnieje udokumentowane potwierdzenie ich istnienia i budowy (*PE=1,* ang. *Protein existance*, wartość 1 ozn. *Evidence at protein level*). W predykacie P3 dodano warunek na numer wersji sekwencji *SV=4*. Predykat P4 wprowadza filtr odnośnie do struktury pierwszorzędowej białka.

Obserwując czasy zapytań SSE1 z dodatkowymi warunkami (rys. 13) i porównując je z czasem wykonania zapytania SSE1 z rysunku 12 można zauważyć, że poprzez zdefiniowanie dobrego filtru złożonego można znacząco skrócić czas potrzebny na realizację zapytania z kilku minut nawet do kilku, jak w przypadku predykatu P3, lub kilkunastu sekund, jak w przypadku predykatu P4. W przypadku analizowanego zapytania warto również zaobserwować (rys. 13) wyraźną korzyść ze zdefiniowania warunków filtrujących w postaci parametru funkcji *sequencePosition* w porównaniu z filtrowaniem przy użyciu klauzuli WHERE.



Rys. 13. Czasy wykonania zapytania ze wzorcem SSE1 oraz dodatkowymi warunkami filtrującymi P1-P4

Fig. 13. Execution times of queries with the SSE1 pattern and additional filtering predicates P1-P4

7. Podsumowanie

Przedstawiony w niniejszym artykule język PSS-SQL pozwala na łatwe wyszukiwanie w bazie danych białek o konstrukcji zbliżonej do zadanego wzorca. Zadany w zapytaniu wzorzec nie musi być ściśle zdefiniowany – pozwala na orientacyjne określenie długości poszczególnych segmentów SSE lub podanie długości nieznanej (lub dowolnej – '*'), typ struktury drugorzędowej może pozostać niezdefiniowany ('?'), segmenty SSE mogą występować opcjonalnie. Prowadzony proces wyszukiwania ma zatem charakter przybliżony, z uwzględnieniem wielu możliwych wariantów dopasowania. Możliwość określenia we wzorcu opcjonalnych segmentów SSE może być potraktowane jako jawne pozwolenie użytkownika na wprowadzenie przerwy w dopasowaniu w konkretnym miejscu.

Integracja metod poszukiwania podobieństwa białek z systemem zarządzania bazą danych daje możliwość łatwego operowania na danych biologicznych bez potrzeby użycia zewnętrznych aplikacji eksploracji danych. Zaproponowane rozszerzenie języka SQL, które zostało przedstawione w niniejszym artykule, jest przykładem takiej integracji. Ma ona wiele zalet. Po pierwsze, logika przetwarzania danych zostaje zdjęta z aplikacji użytkownika i przesunięta w kierunku serwera bazy danych. Zaawansowana analiza danych biologicznych odbywa się zatem podczas pobierania danych z bazy przy użyciu zapytań PSS-SQL, redukując w ten sposób liczbę danych, które zostaną zwrócone z bazy danych oraz ewentualny ruch sieciowy pomiędzy serwerem a aplikacją użytkownika. Po drugie, użytkownicy znający składnię języka SQL bez trudu poradzą sobie z formułowaniem zapytań PSS-SQL. Wprowadzone przez autorów rozszerzenia języka SQL są konstrukcyjnie proste i czytelne, a same zapytania PSS-SQL niezwykle przejrzyste. Daje to przewagę

opracowanemu językowi PSS-SQL nad dotychczas znanymi rozwiązaniami. Za tą prostotą i przejrzystością kryje się jednak wiele operacji transparentnych dla użytkownika, jak choćby przeprowadzenie dopasowania przy użyciu zmodyfikowanego algorytm Smitha-Watermana, należącego do klasy algorytmów programowania dynamicznego. Po trzecie, w rezultacie wykonania zapytań PSS-SQL użytkownicy otrzymują wstępnie przetworzone dane, które następnie mogą wykorzystać do dalszych celów, np. mogą potraktować otrzymane wyniki jako ściśle wyselekcjonowane białka, spełniające podane kryteria odnośnie do budowy przestrzennej, które poddadzą głębszej analizie.

BIBLIOGRAFIA

1. Eidhammer I., Inge J., Taylor W.R..: Protein Bioinformatics: An Algorithmic Approach to Sequence and Structure Analysis. John Wiley & Sons, 2004.
2. Branden C., Tooze J.: Introduction to Protein Structure. Garland 1991.
3. Dickerson R.E., Geis I.: The Structure and Action of Proteins. 2nd ed. Benjamin/Cummings, Redwood City, Calif. Concise 1981.
4. Creighton T.E.: Proteins: Structures and molecular properties. 2nd ed. Freeman, San Francisco 1993.
5. Berman H.M., Westbrook J., Feng Z., Gilliland G., Bhat T.N., Weissig H., et al.: The Protein Data Bank. Nucleic Acids Res., 2000, No. 28, s. 235÷242.
6. Gibrat J.F., Madej T., Bryant S.H.: Surprising similarities in structure comparison. Curr Opin Struct Biol, 6(3), 1996, s. 377÷385.
7. Shapiro J., Brutlag D.: FoldMiner and LOCK 2: protein structure comparison and motif discovery on the web. Nucleic Acids Res., 32, 2004, s. 536÷41.
8. Can T., Wang Y.F.: CTSS: a robust and efficient method for protein structure alignment based on local geometrical and biological features. Proc. of the 2003 IEEE Bioinformatics Conference, 2003, s. 169÷179.
9. Yang J.: Comprehensive description of protein structures using protein folding shape code. Proteins, 71(3), 2008, s. 1497÷518.
10. Mrozek D., Małysiak B.: Searching for Strong Structural Protein Similarities with EAST. Journal of Computer Assisted Mechanics and Engineering Sciences, 2007, No. 14, s. 681÷693.
11. Hammel L., Patel J.M.: Searching on the secondary structure of protein sequences. Proceedings of the 28th international conference on Very Large Data Bases, Hong Kong, China, 2002, s. 634÷645.

12. Tata S., Patel J.M., Friedman J.S., Swaroop A.: Declarative Querying for Biological Sequences. Proc. of the 22nd International Conference on Data Engineering, IEEE Computer Society, 2006, s. 87÷98.
13. Wang Y., Sunderraman R., Tian H.: A Domain Specific Data Management Architecture for Protein Structure Data. Proceedings of the 28th IEEE EMBS Annual International Conference, New York City, USA, IEEE, 2006, s. 5751÷5754.
14. Murzin A.G., Brenner S.E., Hubbard T., Chothia C.: SCOP: A Structural Classification of Proteins Database for the Investigation of Sequences and Structures. J. Mol. Biol. 247, 1995, s. 536÷540.
15. Orengo C.A., Michie A.D., Jones S., et al.: CATH – A hierarchic classification of protein domain structures. Structure, Vol 5. No 8. 1997, s.1093÷1108.
16. Smith T.F., Waterman M.S.: Identification of common molecular subsequences. J Mol Biol, 147, 1981, s. 195÷197.
17. Apweiler R., Bairoch A., Wu C.H., Barker W.C., et al.: UniProt: the Universal Protein knowledgebase. Nucleic Acids Res., 32 (Database issue), 2004, s.115÷9.
18. Frishman D., Argos P.: Incorporation of non-local interactions in protein secondary structure prediction from the amino acid sequence. Protein Eng, 9(2), 1996, s. 133÷142.

Recenzent: Dr Ewa Romuk

Wpłynęło do Redakcji 20 stycznia 2010 r.

Abstract

Protein secondary structures are valuable source of information regarding the construction of the important biological molecules. Secondary structure representation of protein spatial structure allow to study the general shape of proteins and the formation of amino acid chain caused by local and distant hydrogen interactions. The representation gives also the possibility to distinguish types of secondary structures the proteins are built up from – are there only α-helices or only β-strands in the structure? or maybe the structure is generically differentiated. We can also study whether these secondary structure elements are heavily segregated or they appear alternately.

For these reasons, the secondary structure representation of proteins became very important in the analysis of protein constructions and functions. The knowledge of the type of the secondary structure element for consecutive residues in the amino acid chain is frequently

used in the structural similarity searching as one of the phase in the entire process. It decreases the time required to complete the similarity searching, which is a very time-consuming process, since proteins are composed of hundreds of amino acids, and therefore, thousands of atoms linked to each other by covalent bonds.

Protein similarity searching is carried through the comparison of the specified structure to protein structures stored in a database. This is usually done by external applications. Consecutively, data describing protein structures are managed by database management systems (DBMSs), which work excellent in commercial uses. However, they are not dedicated for storing and processing biological data. They do not provide the native support for processing biological data with the use of the SQL language, which is a fundamental, declarative way of data manipulation in most database systems. There are just a few solutions that allow advanced processing of biological data on the database side. However, they are merely prototypes of extensions incorporated into existing DBMSs.

In the paper, we show our extension to the SQL language that allows submission of queries regarding protein spatial structures represented by secondary structure elements. The PSS-SQL language (*Protein Secondary Structure – Structured Query Languaage*) that we have designed and developed supports searching the database against proteins having the secondary structure similar to the structure specified in the user's PSS-SQL query. The query structure is represented by the structural pattern. In the PSS-SQL, we offer a declarative method of searching proteins having particular structural pattern, which is integrated with the database server.

Adresy

Dominika WIECZOREK: studentka Politechniki Śląskiej, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, wieczorek.dominika@gmail.com .
Bożena MAŁYSIAK-MROZEK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, bozena.malysiak@polsl.pl .
Dariusz MROZEK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, dariusz.mrozek@polsl.pl .