Seminarium z przedmiotu: Algorytmy Genetyczne

Metody wzorowane na naturze:

- Obliczenia DNA.
- Sieci neuronowe.

Kamila Folta
Nr albumu: 96411
Wydział: Informatyka i Zarządzanie
Kierunek: Informatyka
Specjalność: Inżynieria Oprogramowania
Profil: Sztuczna Inteligencja, Bazy Danych

Obliczenia DNA.

DNA – kwas dezoksyrybonukleinowy; koduje geny i większość, jeśli nie wszystkie, operacji na komórkach, przy użyciu czteroliterowego alfabetu zasad: A – adenina, T – tymina, C – cytozyna, G – guanina. Jako długi łańcuch zasadowy, DNA wykazuje pewne podobieństwo do taśmy maszyny Turinga. Niestety, wiele prostych i pożądanych operacji na takich łańcuchach nie może być przeprowadzonych z chemicznego punktu widzenia.

Z drugiej strony, wiele innych prostych operacji jest możliwych. DNA jest w naturalnym stanie złożone z dwóch nici, owijających się wokół siebie, w postaci helisy. Dwie nici są utrzymywane razem dzięki wiązaniom między odpowiadającymi sobie zasadami: A łączy się z T oraz C z G. W rezultacie, sekwencja zasad na jednej z nici jest komplementarna z sekwencją na drugiej nici. Na przykład, ATTGTC połączy się z łańcuchem komplementarnym TAACAG, tworząc cząsteczkę podwójną.

Polimeraza DNA.

Polimeraza DNA może być rozumiana jako nanomaszyna: jest to cząsteczka, która przechodząc przez kolejne zasady nukleinowe na jednej nici, a więc czytając podstawowe informacje, odtwarza jednocześnie ciąg zasad komplementarnych, czyli tworzy dopełnienie kodu genetycznego zawartego na nici DNA. Istota mechanizmu polimerazy jest zadziwiająco podobna do opisanej w 1936 roku przez Alana M. Turinga, idei maszyny wielostanowej, nazwanej później maszyną Turinga. Rozpoczęte wówczas badania, prowadzone niezależnie przez Kurta Gödla, Alonzo Churcha i S.C. Kleene'a, dotyczyły zagadnień obliczalności, wyprzedziły o dziesięciolecie powstanie komputerów oraz doprowadziły do kilku najbardziej znaczących osiągnięć matematyki XX wieku [patrz: Howard DeLong, „Unsolved Problems in Arithmetics", Scientific American, marzec 1971 oraz Gregory J. Chaitin, „Randomness in Arithmetics", SA, lipiec, 1988].

Maszyna Turinga, jako model koncepcyjny, była niezwykle prosta: składała się z dwóch taśm i kontrolera o skończonej liczbie stanów, który porusza się po taśmie z danymi wejściowymi i jednocześnie przewija taśmę z danymi wyjściowymi, pisząc na niej i ją odczytując. Kontroler może wykonywać programy, napisane przy pomocy prostych instrukcji i łatwo byłoby napisać taki program, który pobiera z taśmy ciąg znaków A, T, G, C i wypisuje na taśmie z danymi wyjściowymi komplementarny ciąg, tak, jak się to odbywa w odkrytym przez Watsona i Cricka mechanizmie. Podobieństwo do polimerazy DNA jest uderzające.

Istnieje jeszcze jeden fakt, który uczynił ideę obliczeń DNA bardziej intrygującą. Teza Churcha – Turinga mówi o uniwersalności TM: dowolne obliczenie, które w ogóle da się przeprowadzić, może być wykonane na odpowiednio zaprogramowanej maszynie Turinga. Koncepcja maszyny bazującej na DNA stała się inspiracją dla Leonarda Adlemana: próbował on znaleźć taki enzym lub grupę enzymów, który mógłby pełnić rolę kontrolera w maszynie DNA. Prawie 10 lat wcześniej na ten sam pomysł wpadli Charles H. Benner i Rolf Landauer z IBM [patrz: „The Fundamental Physical Limits of Computation"; Scientific Americain, lipiec, 1985]. Niestety, choć znany jest enzym zdolny syntezować komplementarną nić DNA (polimeraza DNA), nie wydawało się prawdopodobne, że istnieją enzymy zdolne do rozkładu liczb zakodowanych przy pomocy zasad nukleinowych na czynniki pierwsze.

Dostępne operacje na DNA.

1. Parowanie się nici komplementarnych.

Każda nić dna, która napotka w roztworze swoją nić komplementarną, połączy się z nią, tworząc słynną helisę (podwójną spiralę). Nici nie łączą się wiązaniami kowalencyjnymi, lecz są utrzymywane ze sobą słabszymi wiązaniami wodorowymi. Nić DNA nie sparuje się z nicią, która nie posiada długiego odcinka komplementarnego.

2. Polimerazy.

Enzymy te kopiują informacje z jednej cząsteczki na drugą. Polimeraza DNA tworzy nić komplementarną do nici – matrycy. Potrzebny jest jednak pewien sygnał startowy dla mechanizmu kopiowana, będący informacją o miejscu, od którego ma rozpocząć się kopiowanie. Jest on zapewniony przez tzw. primer – specyficzną nić DNA, często bardzo krótką, która paruje się z odpowiednim miejscem na nici matrycy. Polimeraza DNA rozpoczyna kopiowanie od miejsca, na którym napotka kompleks matryca – primer. Jednokrotna operacja polimerazy podwaja liczbą kopii; wielokrotne zastosowanie reakcji polimerazy powoduje ekspotencjalny przyrost liczby kopii.

3. Ligazy.

Te enzymy łączą ze sobą cząsteczki. Na przykład Ligaza DNA może połączyć wiązaniem kowalencyjnym dwie pobliskie nici DNA w jedną dłuższą. Komórki posługują się ligazą DNA do reperowania uszkodzeń DNA, powstałych w komórkach na przykład w wyniku działania promieni ultrafioletu światła słonecznego ma komórki skóry.

4. Nukleazy.

Innym mechanizmem zastosowanym przez naturę jest użycie enzymów restrykcyjnych, których zadaniem jest przecinanie obcych łańcuchów DNA zawierających szczególny wzorzec. Nukleazy tną kwasy nukleinowe. Na przykład endonukleazy restrykcyjne przeglądają nić DNA, szukając specyficznej sekwencji zasad, a po jej znalezieniu rozcinają nić w tym miejscu. Enzym EcoRI (z bakterii Escherichia coli) to enzym restrykcyjny – restryktaza – który rozetnie nić DNA po G w sekwencji GAATTC. Prawie nigdy nie rozcina on nici w innym miejscu. Enzymy nie przecinają helisy prosto; zostawiają na końcach „wiszące fragmenty", dzięki którym możliwe jest przyłączenie właściwych dla bakterii, opowiednich wzorców DNA, a także – parząc z punktu widzenia operacji na łańcuchu – przyłączenie odpowiednich fragmentów rozwiązań. Niestety, enzymy RE (restriction enzymes) rzadko rozpoznają sekwencje dłuższe niż sześciozasadowe. Podejrzewano, że restryktazy wyewoluowały u bakterii jako mechanizm obronny przed wirusami. Bakteria E.coli umie chronić swoje DNA przed działaniem enzymu EcoRI, ale DNA wirusowe zostanie zniszczone.

5. Elektroforeza na żelu.

Mechanizm nie występuje, tak jak opisane wyżej, w warunkach naturalnych. Polega na umieszczeniu roztworu różnych nici DNA z jednej strony płytki specjalnego żelu, przez ktory przepuszczamy prąd. Cząsteczki DNA, któr są naładowane ujemnie, dążą do anody; krótsze nici poruszają się szybciej od długich. W rezultacie cząsteczki DNA zostają rozdzielone według długości. Dzięki specjalnym odczynnikom można zobaczyć w ultrafiolecie, dokąd cząsteczki o różnej długości dotarły.

6. Synteza DNA.

Jest to sztuczne otrzymywanie cząsteczek DNA kodujących zadaną z góry sekwencję zasad A, T, C, G. Istnieją obecnie komercyjne laboratoria, które wykonują syntezę DNA na zlecenie, po uzyskaniu od klienta żądanej sekwencji. Proces taki trwa kilka dni, po czym otrzymuje się probówkę zawierającą ok. 10 exp 18 cząsteczek DNA, z których prawie wszystkie mają żądaną sekwencję zasad. Obecnie można w ten spsób otrzymywać sekwencje złożone z ok. 100 zasad. DNA 20-zasadowe kosztuje ok 25 dolarów. Jest dostarczane suche w probówce i ma postać białego pyłu.

Technologia rekombinacyjna DNA oparta jest na takich operacjach: przecinaniu DNA i pozwalaniu wolnym końcom na przyłączanie się partnerskim niciom oraz kopiowaniu.

Ogromna liczba cząteczek DNA pozwala na pojawienie się ogromnej ilości możliwych kombinacji, mogących zarówno reprezentować zupełnie losowe liczby, jak również mieć semantykę rozwiązania problemu NP – zupełnego. Podsumowując, są możliwe następujące operacje do przeprowadzenia na DNA: (1) duplikacja i zwielokrotnienie nici; (2) tworzenie nici o określonej długości; (3) rekombinacja na wielką skalę – przy pomocy końcówek komplementarnych, do których mogą przyłączać się inne nici oraz (4) separacja na podstawie długości.

Problemy NP – zupełne.

Ponieważ DNA ma naturę liniową, a zestaw narzędzi do manipulowania na nim jest raczej ograniczony, pewne problemy są w naturalny sposób bardziej odpowiednie do rozwiązywania przy jego pomocy. Można mieć pewność, że każdy łańcuch może być przedstawiony w czteroliterowym alfabecie DNA. Jeśli na przykład zakodujemy dwie liczby całkowite przy pomocy łańcuchów czterozasadowych, to w jaki sposób otrzymać łańcuch będący reprezentacją ich iloczynu? Takie pytanie wcale nie jest trywialne, zarówno na poziomie abstrakcyjnym (wiedząc, jakie operacje na łańcuchach są dopuszczalne), jak i pod względem chemicznym – uwzględniającym możliwości laboratoriów badawczych. Jednak przynajmniej w teorii, stworzenie komputera uniwersalnego wymaga dwu rzeczy: sposobu zapisu informacji oraz paru nieskomplikowanych operacji na danych – zestawu instrukcji. Te zaś są zapewnione.

Problem Hamiltona, rozwiązanie Adlemana.

Wiliam Rowan Hamilton był królewskim astronomem w Irlandii w XIX wieku. Problem noszący jego imię polega na rozstrzygnięciu, czy dla danego grafu z określonymi wierzchołkami początkowym i końcowym istnieje droga Hamiltona. Droga Hamiltona między wierzchołkiem początkowym i końcowym istnieje wtedy i tylko wtedy, gdy istnieje droga zaczynająca się w wierzchołku początkowym i kończąca się w docelowym, przechodząca przez każdy z pozostałych wierzchołków dokładnie raz.

Zagadnienie drogi Hamiltona, badane intensywnie przez informatyków, okazało się być problemem NP-zupełnym (udowodniono to na początku lat siedemdziesiątych), co potwierdziło przekonania informatyków o niemożności znalezienia efektywnego algorytmu, mimo że zagadnienie „NP = P?” jest do dziś jednym z bardziej doniosłych

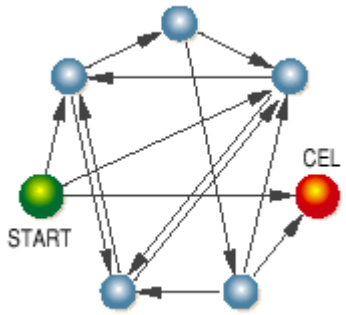
nieudowodnionych hipotez [patrz: John E. Hopcroft, „Turing Machines", Scientific American, maj 1984].

Zagadnienie drogi Hamiltona stanowiło wyzwanie dla Leonarda Adlemana, który do swoich pierwszych obliczeń DNA zastosował następujący algorytm"

Dla grafu z n wierzchołkami:

1. **Utwórz zbiór losowych dróg, przechodzących przez graf.**
2. **Dla każdej drogi sprawdź, czy:**
    a. **zaczyna się w wierzchołku początkowym i kończy w docelowym; jeśli nie, usuń ją ze zbioru;**
    b. **przechodzi dokładnie przez n wierzchołków; jeśli nie, usuń ją ze zbioru;**
    c. **przechodzi przez każdy wierzchołek; jeśli nie, usuń ją ze zbioru.**
3. **Jeśli powstały zbiór jest niepusty, to istnieje droga Hamiltona, w przeciwnym razie – nie istnieje.**

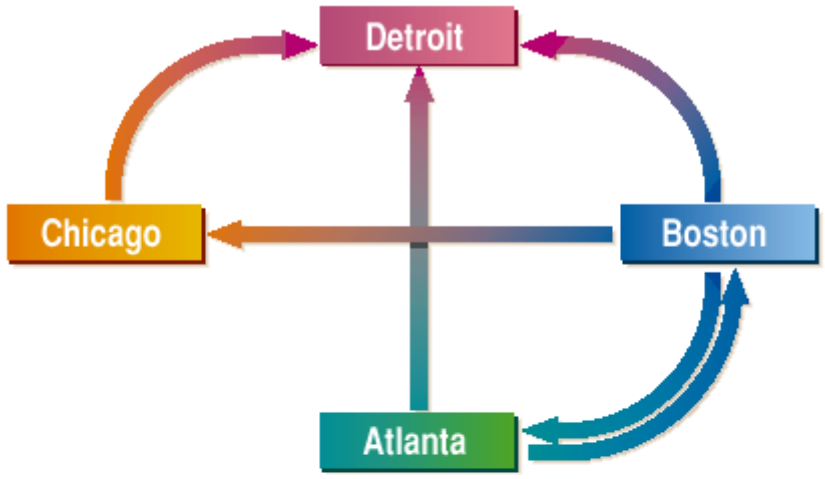
Nie jest to algorytm idealny. Prawdopodobieństwo uzyskania prawidłowej odpowiedzi zależy od tego, czy zbiór dróg jest dostatecznie losowy i dostatecznie liczny. Na potrzeby eksperymentu z obliczeniami DNA został wybrany problem na tyle prosty, by w warunkach laboratoryjnych mógł być przeprowadzony, na tyle jednak złożony, by mógł stanowić dowód działania komputera DNA. Adleman wybrał mapę obejmującą siedem miast i czternaście dróg.



Przykład.

W przedstawionym przykładzie zredukowano problem do czterech miast oraz sześciu połączeń między nimi. Rozpatrzmy miasta: Atlantę, Boston, Chocago i Detroit, połączonych sześcioma lotami. Problem polega na znalezieniu drogi Hamiltona zaczynającej się

w Atlancie, a kończącej się w Detroit. Każdemu miastu przyporządkować można losowe ośmioelementowe sekwencje DNA: Atlancie – ACTTGCAG, Bostonowi – TCGGACTG itd.



MIASTO	DNA	DOPEŁNIENIE
ATLANTA	ACTTGCAG	TGAACGTC
BOSTON	TCGGACTG	AGCCTGAC
CHICAGO	GGCTATGT	CCGATACA
DETROIT	CCGAGCAA	GGCTCGTT

LOT	DNA
ATLANTA–BOSTON	GCAGTCGG
ATLANTA–DETROIT	GCAGCCGA
BOSTON–CHICAGO	ACTGGGCT
BOSTON–DETROIT	ACTGCCGA
BOSTON–ATLANTA	ACTGACTT
CHICAGO–DETROIT	ATGTCCGA

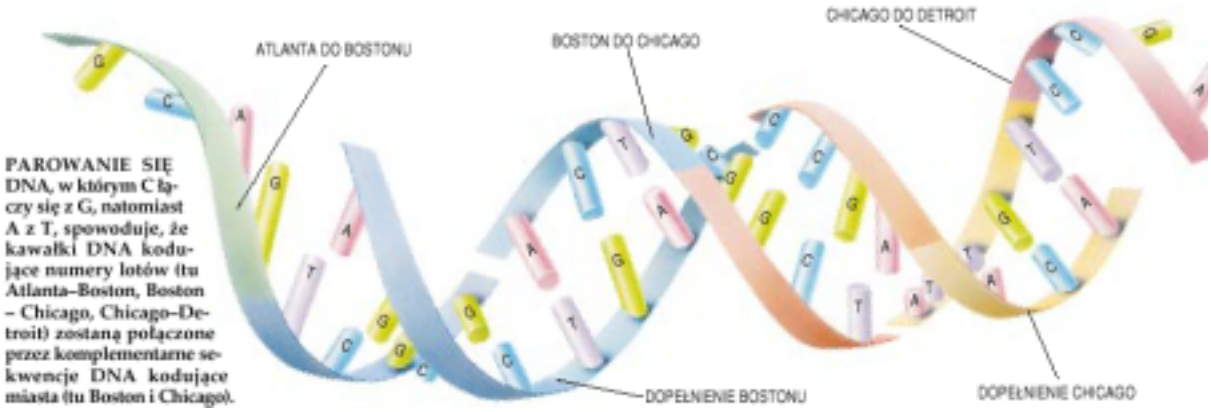
SUM FILMS

Niech pierwsze cztery zasady kodujące miasto oznaczają jego imię, następne cztery zaś – nazwisko. Konieczne jest też zakodowanie tras między miastami. Każdy bezpośredni odcinek otrzymuje symbol będący złożeniem nazwiska miasta startowego i imienia miasta docelowego; trasa między Atlantą i Bostonem ma postać GCAGTCGG. Każdej sekwencji DNA odpowiada sekwencja komplementarna, a więc kodowi każdego z miast będzie odpowiadał kod komplementarny, np. dla Atlanty: TGAACGTC. Do przeprowadzenia eksperymentu potrzebne były łańcuchy komplementarne dla każdego miasta i każdego połączenia, które Adleman otrzymał na drodze syntezy DNA. Eksperyment polegał na wrzuceniu szczypty (ok. 10 exp 14 cząsteczek) każdego z otrzymanych DNA do probówki oraz dodaniu wody z odrobiną ligazy, soli i innych składników, koniecznych do odtworzenia w przybliżeniu warunków panujących wewnątrz komórki. W sumie potrzebny roztwór

zajmował objętość ok. jednej pięćdziesiątej łyżeczki do herbaty. Po ok. jednej sekundzie w probówce znajdowało się rozwiązanie zagadnienia Hamiltona.

Co działo się w probówce?

Nić kodująca połączenie Atlanta – Boston (GCAGTCGG) i komplementarny symbol Bostonu (AGCCTGAC) mogły się przypadkowo spotkać. Nić kodująca połączenie kończy się sekwencją TCGG, a komplementarny symbol Bostonu zaczyna się od sekwencji AGCC. Jako sekwencje komplementarne, przylgną one do siebie. Jeśli tak utworzony kompleks napotka nić odpowiadającą trasie Boston – Chicago (ACTGGGCT), to ją przyłączy, ponieważ jej początek (ACTG) jest komplementarny z końcem kompleksu (TGAC). W ten sposób kompleks będzie się wydłużał – nici komplementarne odpowiadające miastom będą spinały odpowiednie sekwencje kodujące połączenia między nimi. Ligaza zawarta w probówce połączy trwale powstałe kompleksy. Powstaną więc nici podwójnego DNA, kodujące różne dopuszczalne loty przez różne miasta.



PAROWANIE SIĘ DNA, w którym C łączy się z G, natomiast A z T, spowoduje, że kawałki DNA kodujące numery lotów (tu Atlanta–Boston, Boston – Chicago, Chicago–Detroit) zostaną połączone przez komplementarne sekwencje DNA kodujące miasta (tu Boston i Chicago).

Ze względu na ogromną (w porównaniu z liczbą rozpatrywanych miast) liczbę cząsteczek DNA w probówce, można było mieć pewność, że jedna z powstałych cząsteczek koduje drogę Hamiltona. Wszystkie drogi powstały w tym samym czasie w wyniku jednoczesnej interakcji setek bilionów cząsteczek. Reakcja biochemiczna odpowiadała w tym sensie wielotorowemu przetwarzaniu równoległemu.

Cząsteczka kodująca rozwiązanie w przykładzie ma sekwencję GCAGTCGGACTGGGCTATGTCCGA.

Niestety, probówka, oprócz poszukiwanego rozwiązania, zawierała około 100 bln cząsteczek, które nie kodowały drogi Hamiltona. Proces eliminowania tych rozwiązań rozpoczął się od wyselekcjonowania rozwiązań zaczynających i kończących się w odpowiednio startowym i końcowym mieście, przy pomocy techniki PCR (polimerase

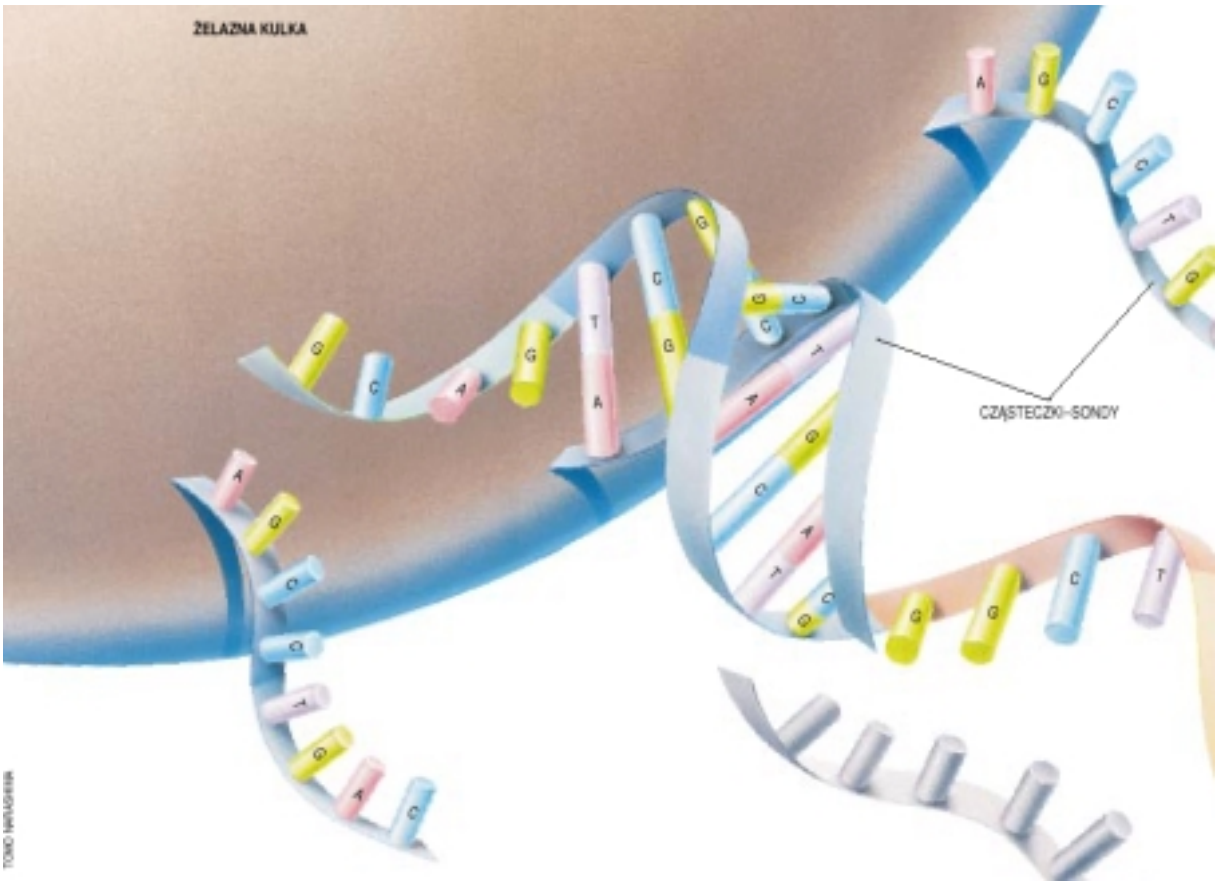
chain reaction – łańcuchowa reakcja polimerazy). Wymagała ona wielu kopii dwóch krótkich kawałków DNA, mających posłużyć jako primery dla polimerazy DNA: primera, kodującego drugą część (nazwisko) miasta początkowego (GCAG odpowiadający Atlancie) oraz kodującego komplementarny kod miasta docelowego (GGCT - Detroit). Pierwsza nich nakazywała polimerazie wytwarzać komplementarne kopie sekwencji, które zaczynały się kodem odpowiedniego miasta początkowego, druga natomiast rozpoczynała namnażanie cząsteczek, które zawierały kod właściwego miasta docelowego.

PCR polega na wytwarzaniu cykli cieplnych (*thermocycling*) – wielokrotnym podnoszeniu i obniżaniu temperatury mieszanki w probówce. W wyniku podgrzania polimeraza DNA zaczyna powielać cząsteczki DNA, w jeszcze wyższej temperaturze wytworzone podwójne helisy DNA rozszczepiają się na pojedyncze nici, co pozwala na kolejną replikację w następnej fazie.

W wyniku tej procedury cząsteczki o właściwym kodzie początkowym i końcowym namnażały się wykładniczo. W przeciwieństwie do nich, cząsteczki, które miały dobry tylko kod miasta początkowego, a zły kod miasta końcowego lub dobry tylko kod końcowy, namnażały się liniowo. Pozostałe cząsteczki nie namnażały się wcale. W wyniku działania PCR powstał płyn, w którym było bardzo dużo cząstek zawierających zarówno właściwy kod początkowy i końcowy, i nieliczne łańcuchy nie spełniające tego kryterium. W ten sposób został wykonany krok 2a algorytmu.

Aby wyselekcjonować cząsteczki właściwej długości (w rozpatrywanym przykładzie powinny mieć długość 24 zasad), należało posłużyć się elektroforezą. Był to krok 2b algorytmu.

Aby sprawdzić, czy wyselekcjonowane cząsteczki zawierają kody wszystkich miast, przez które ma przechodzić droga, zastosowano technikę zwaną rozdziałem opartym na powinowactwie (*affinity separation*).



W procesie tym używa się wielu kopii „próbkującej" cząsteczki DNA, tzw. sondy, która koduje komplementarną sekwencję danego miasta. Próbkujące DNA doczepia się do mikroskopijnych stalowych kuleczek o średnicy mniej więcej jednego mikrometra. Kuleczki umieszcza się w warunkach sprzyjających parowaniu się łańcuchów DNA. Cząsteczki zawierające kod poszukiwanego miasta (np. Bostonu) wiązały się z DNA próbkującym (sondą). Aby oddzielić je od reszty, do brzegu probówki wystarczyło przyłożyć magnes, który przyciągnął stalowe kuleczki – resztę płynu można było wylać. Po dodaniu nowego rozpuszczalnika i usunięciu pola magnetycznego, kuleczki zostały uwolnione. Zwiększenie temperatury spowodowało oderwanie się DNA od kuleczek i przejście do roztworu. Ponowne przyłożenie magnesu przyciągnęło kuleczki – tym razem bez nici DNA. Roztwór zawierał cząsteczki kodujące drogi, które przechodzą przez Boston. Procedura była powtórzona dla pozostałych miast tranzytowych. Zajęła ona Adlemanowi cały dzień i była „najbardziej żmudną częścią doświadczenia".

Dzięki rozdziałowi opartemu na powinowactwie stanowiącemu krok 2c algorytmu, w probówce powinny były pozostać cząsteczki kodujące drogę Hamiltona. Jeśli więc w roztworze były jeszcze jakieś nici DNA, oznaczało to, że istnieje droga Hamiltona dla

danego grafu; jeśli nie było żadnego DNA, oznaczałoby to, że droga taka nie istnieje. Do sprawdzenia wyniku powtórnie została zastosowana technika PCR, a potem analiza elektroforezą na żelu. Końcowa analiza wykazała że pozostałe w probówce DNA kodowało poszukiwaną drogę Hamiltona. Doświadczenie Adlemana zakończyło się po siedmiu dniach spędzonych w laboratorium. Pierwsze obliczenia za pomocą DNA zakończyły się sukcesem.

Nowa dziedzina.

Rezultatem coraz większej kontroli nad światem molekularnym jest nowa dziedzina wiedzy. Geralt F. Joyce zs Scripps Research Institute w La Jolla (Kalifornia) „hoduje ” biliony cząsteczek RNA, generacja po generacji, dążąc do otrzymania superpopulacji - takich supercząsteczek, które będą posiadały poszukiwane własności katalityczne [patrz: Geralt F. Joyce, „*Ukierunkowana ewolucja molekularna*”, Świat Nauki, luty 1993]. Julius Rebck, Jr, z MIT tworzy cząsteczki które potrafią się replikować, co ma umożliwić wgląd w początki życia na Ziemi [patrz: Julius Rebek, Jr, „*Samoreplikujące się cząsteczki chemiczne*”; Świat Nauki, wrzesień 1994]. Erick Winfree z California Institute of Technology zaczął syntetyzować „inteligentne” kompleksy molekularne, dające się zaprogramować tak, aby łączyły się w z góry zadane struktury o dowolnej złożoności.

Już wcześniej potrafiono znajdować ścieżki Hamiltona w grafach o podobnych do rozważanego przez Adlemana wymiarach i zwykłe metody były znacznie szybsze. Teoretyczna analiza złożoności tego eksperymentu wskazuje jednak, że rozmiar problemu i czas potrzebnych obliczeń są w dobrej proporcji. Procedura Adlemana używa wielomianowego czasu przetwarzania, ale obecnie znane algorytmy potrzebują aż wykładniczego czasu przetwarzania na maszynach sekwencyjnych, czego nie można uznać za szczególnie efektywne. Duża równoległość powoduje dużą oszczędność czasową, przy dostatecznie dużym rozmiarze danych wejściowych. Uogólniając metodę Adlemana, uzyskuje się równoległy model przetwarzania, w którym cząsteczki są rozproszonymi zbiorami danych, natomiast manipulacje na cząsteczkach odpowiadają operacjom rozproszonym. W tubie testowej można manipulować milionami cząsteczek, podczas gdy krzemowe komputery równoległe są obecnie ograniczone do tysięcy procesorów. Pojawia się pytanie o możliwość przekroczenia „wykładniczej granicy” i, co byłoby naturalną tego konsekwencją, rozwiązania w rozsądnym czasie problemów nierozwiązywalnych w praktyce (ponieważ zajęłoby to

tysiące lat). Jednak już proste obliczenia wskazują, że procedura Adlemana wymagałaby więcej cząsteczek, niż zawiera ich cała Ziemia, dla obliczeń ścieżki Hamiltona w grafie z dwustoma wierzchołkami. Informatyka teoretyczna zajmuje się obecnie znajdowaniem ograniczeń i rozpatrywaniem ewentualnych obszarów zastosowań dla nowego modelu obliczeń równoległych.

Reakcje na eksperyment Adlemana.

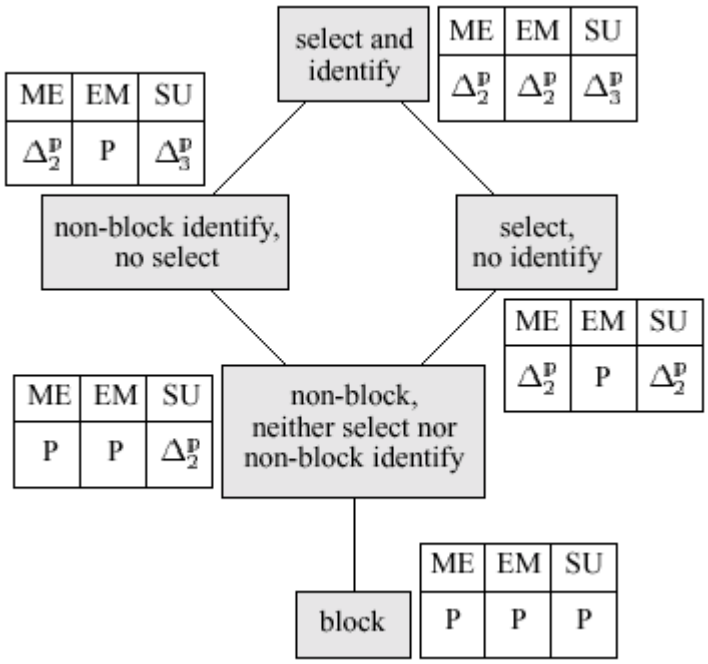
Jedną z pierwszych osób, które zajęły się tworzeniem modelu dla rezultatów obliczeń DNA, przeprowadzonych przez Adlemana, był Lipton. Obierając za punkt wyjścia kilka wyidealizowanych założeń, otrzymał deterministyczny model przetwarzania równoległego. Założył, że na DNA znajdującym się w tubie testowej można przeprowadzać operacje molekularne: łączenie (*union*), inicjalizację (*initialization*), ekstrakcję (*extraction*), zwielokrotnianie (*amplification*) oraz test nieistnienia (*emptiness test*). Na tej podstawie zbudował model obliczenia molekularnego, wykonanego przez Adlemana. Jednym z ważniejszych osiągnięć był dowód, że każdy problem NP może być przy pomocy tego modelu rozwiązany w czasie wielomianowym.

Rozszerzeniem tego modelu były prace Roossa i Wagnera, którzy dodali inne operacje i testy realizowalne na DNA. Operacje te mogą być podzielone następująco:

1. Operacje, mające własność blokowania: zapewniają, że albo wszystkie nici w tubie mają określoną długość, albo wszystkie mają długość różną od określonej.
2. Operacje, mające własność selekcji: działające na nici o szczególnych własnościach (modyfikujące), nie zmieniające nici nie posiadających takich własności.
3. Operacje, mające własność identyfikacji: zapewniają rozróżnianie nici, na przykład poprzez identyfikację lub cięcie określonych sekwencji czy podłańcuchów.

Operacje, mające własności zarówno identyfikacji jak i blokowania nazwano operacjami prostej identyfikacji. Lipton używa testu nieistnienia (*emptiness test*, EM), który daje odpowiedź pozytywną, gdy w probówce nie ma nici spełniających pewne założenia. Dodatkowe testy sprawdzają inne własności: test przynależności (*membership test*, ME) daje odpowiedź pozytywną, gdy w probówce jest nić spełniająca założenia, oraz test podzbioru (*subset test*, SU), który jest pozytywny, gdy w probówce znajduje się pewien podzbiór zbioru spełniającego założenia.

Aby skonstruować algorytm w tym modelu, możliwe jest podanie zarówno zbioru instrukcji w klasycznym języku programowania, np. w Pascalu, jak i zestawu operacji molekularnych. Rozpatruje się obecnie różne warianty takich języków programowania „DNA-Pascal", które różnią się zestawem dopuszczalnych molekularnych operacji i testów. Jeśli ograniczymy zastosowanie tych języków do problemów wymagających czasu wielomianowego , zdefiniują one nową klasę problemów, nie występującą w klasycznych rozważaniach nad złożonością.



Jeśli używamy tylko operacji blokowania, żaden test molekularny nie zwiększy mocy języka DNA-Pascal z wielomianowym ograniczeniem czasowym powyżej P (wielomianowe). Operacje bez blokowania i selekcji, ale z co najwyżej prostą identyfikacją zwiększają czas DNA-Pascala do wielkości P exp NP. Operacje selekcji zwiększają możliwości testu EM, jak również operacje identyfikacji bez blokowania, przy użyciu testów ME i SU. Taki sam efekt daje zastosowanie prostej identyfikacji z operacjami selekcji.

Komputer uniwersalny.

Pojęcie komputera uniwersalnego odnosi się do maszyny potrafiącej rozwiązać dowolny problem obliczeniowy. Na wejściu dostaje on program będący implementacją algorytmu mającego problem rozwiązać, oraz parametry potrzebne w tym algorytmie – dane wejściowe. Istnieje wiele równoważnych modeli obliczeniowych reprezentujących komputer uniwersalny, z których prawdopodobnie najważniejszym jest maszyna Turinga. Każde

obliczenie w tej maszynie jest reprezentowane jako sekwencja stanów. Każdy stan koduje pojedynczy krok obliczeniowy. Kolejne stany różnią się od siebie w lokalnie ograniczonej przestrzeni. Maszyna Turinga akceptuje ciąg wejściowy, jeśli istnieje sekwencja konfiguracji (stanów), która zaczyna się od stanu początkowego i kończy na stanie końcowym. Teza Churcha mówi, że każde obliczenie, które jest możliwe do wykonania, może być przeprowadzone na maszynie uniwersalnej. Według tego standardu każdy nowy model obliczeniowy musi być przetestowany czy jest maszyną uniwersalną, czy nie.

Beaver opisuje uniwersalny komputer molekularny. Symuluje maszynę Turinga poprzez zakodowanie wszystkich jej stanów w łańcuchach DNA. Koduje wszystkie pary konfiguracji poprzednik – następnik. W wyniku rekombinacji DNA powstają coraz dłuższe łańcuchy DNA. W rezultacie wystarczy sprawdzić, czy istnieje łańcuch kodujący stan początkowy na początku, później kodowane są dowolne stany oraz stan końcowy (akceptujący) na końcu. Udowodniono, że model Liptona może przetwarzać dowolny problem obliczeniowy.

Komputery molekularne.

Molekularne komputery mają wiele atrakcyjnych własności. Pozwalają na niezmiernie gęste upakowanie informacji. Na przykład gram DNA, który w suchej postaci zajmuje ok. 1 cm^3, może przechować tyle informacji, ile daje się zapisać na bilionie CD-romów. Komputery molekularne zapewniają niezwykle duży stopień równoległości przetwarzania. W objętości 1/50 łyżeczki od herbaty zostało w ciągu 1s zsyntezowanych około 10 exp 14 kodujących różne drogi cząsteczek DNA.

Komputery molekularne są potencjalnie bardzo energooszczędne. W zasadzie 1J wystarcza do wykonania 2 * 10 exp 19 operacji łączenia cząsteczek DNA. Jest to dużo, jeśli weźmie się pod uwagę, że według drugiego prawa termodynamiki maksymalna liczba (nieodwracalnych) operacji możliwych do wykonania za pomocą energii 1J wynosi 24 * 10 exp 19. Dzisiejsze superkomputery są znacznie mniej wydajne: dżul wystarcza na najwyżej 10 exp 9 operacji.

Problem obwodów Boole'a.

Obwody boolowskie stanowią jeden z najgłębiej przeanalizowanych i przestudiowanych modeli przetwarzania równoległego. Jednostkami obliczeniowymi

są w nim bramki, które wykonują funkcje „not”, „and” i „or”. Złożoność tych obwodów dana jest przez parametry *fan-in* i *fan-out* (maksymalną liczbę odpowiednio wejść i wyjść), rozmiar (*size*) – liczbę bramek oraz głębokość (*depth*) – najdłuższą ścieżkę w obwodzie. Często liczba wejść w bramkach typu „and” jest ograniczona do dwóch, podczas gdy liczba wejść w bramkach „or” jest ustalona arbitralnie. Takie obwody są nazywane pół – nieograniczone (*semi - unbounded*). Symulacja obwodów boolowskich przy pomocy komputera DNA była intensywnie badana. Ogihara i Ray podali algorytm molekularny, który symuluje pół – nieograniczone obwody boolowskie o głębokości t i rozmiarze g w czasie $O(t \log f)$ przy wykorzystaniu DNA rzędu $O(gf)$, gdzie f oznacza liczbę wyjść z obwodu. W szczególnym przypadku uzyskane wyniki oznaczają, że jest możliwe przeprowadzenie symulacji dla ważnej klasy obwodów (NC) w czasie liniowym względem głębokości (*depth*) obwodów.

Cząsteczki DNA w roztworze mają swój biochemiczny odpowiednik na powierzchniach stałych. Została rozwinięta związana z tym metoda badawcza oraz model przetwarzania, na którym można efektywnie symulować obwody (liczba operacji równoległych jest proporcjonalna do rozmiaru obwodu).

Kryptografia, złożoność, wnioski.

Zgrubne kalkulacje pokazują, że używane obecnie metody rozwiązywania problemów NP – zupełnych nie powinny wzbudzać strachu w dziedzinie kryptografii, przynajmniej w najbliższej przyszłości. Przypuśćmy, że chcemy znaleźć rozkład na czynniki liczby 1000 – bitowej. Jest to zadanie nierozwiązywalne dla współczesnych komputerów. Używając standardowej redukcji problemu do Ścieżek Hamiltona, otrzymujemy graf z przynajmniej 10^6 (10 exp 6) węzłami. Bezpiecznie przeceniając tę liczbę, mamy 2 exp (10 exp 6) ścieżek między tymi węzłami. Potrzeba więc aż około 10 exp 300000 cząsteczek reprezentujących możliwe rozwiązania. Jeśli przyjmiemy, że rozmiar cząsteczki DNA jest porównywalny z cząsteczką wody (grube niedoszacowanie), wówczas będzie potrzeba 10 exp 200000 litrów DNA z możliwymi rozwiązaniami.

Model Adlemana z pewnymi dodatkowymi operacjami został zastosowany do złamania standardu kodowania danych DES (*data encryption standard*). DES koduje informacje kluczem 56 bitowym. Jeśli będzie dany tekst zaszyfrowany i tekst jawny, wówczas klasyczny algorytm musi przejrzeć sekwencyjnie 2 exp 56 możliwych kluczy, co zabierze mu około dziesięciu tysięcy lat, zakładając wydajność sprzętową stu tysięcy operacji na sekundę. Boneth, Dunworth i Lipton pokazali, jak znaleźć 56 – bitowy klucz

w laboratorium, podczas około czterech miesięcy pracy. Model „*sticker*”, zaproponowany przez Roweisa i Winfree jest modelem przetwarzania, w którym połączono manipulacje na DNA oraz wykorzystano pamięć o losowym dostępie (*Random Access Memory*). Jeden gram DNA wystarcza do złamania kodu DES w rozsądnym czasie.

Komputery molekularne są w stanie rozwiązać problem najkrótszego wspólnego nadłańcucha, choć już oszacowanie jego długości jest problemem obliczeniowo trudnym. Aby sprawdzić, czy istnieje nadłańcuch o długości k taki, że zawiera on wszystkie spośród łańcuchów s1, s2, ..., sn, wystarczy zsyntezować wszystkie możliwe kombinacje liter z rozważanego alfabetu o długości k, a następnie zastosować metodę tekstowego wstawiania / usuwania, aby odfiltrować takie łańcuchy, które zawierają wszystkie podłańcuchy s1,..., sn.

Odporność na błędy.

Większość opisanych modeli opiera się na wyidealizowanych założeniach, w szczególności zakłada się determinizm metod badawczych dotyczących operacji molekularnych i testów. W rzeczywistości, wielokrotne powtarzano eksperyment Adlemana i nie otrzymywano jednoznacznych, wyraźnych rezultatów. Analiza procesów biochemicznych i owa „niepowtarzalność” eksperymentu wskazują, że trudno uznać owe założenia za realistyczne. Górna granica modeli przetwarzania molekularnego z doskonałymi operacjami i testami określa jednocześnie granicę niemożliwą do osiągnięcia przez realistyczną implementację. Jednak głębokie studia nad zagadnieniami analizy błędów i strategii ich unikania pomaga w pełniejszej klasyfikacji rozważanych modeli i pozwala na określenie stopnia realizowalności tych modeli.

Implementacja operacji ekstrakcji przy pomocy reakcji polimerazy (PCR) rzuca cień wątpliwości na prawdziwość wyniku obliczeń. Jeśli założymy, że PCR wyselekcjonuje prawidłowe nici z prawdopodobieństwem 95%, to po przeprowadzeniu stu kroków obliczeniowych, prawdopodobieństwo tego, że w probówce zostaną tylko nici spełniające żądane kryteria wynosi już tylko 0,006%. Amos, Gibbons i Hodgeson zaimplementowali operację ekstrakcji za pomocą enzymów restrykcyjnych, które niszczą wszystkie nici zawierające określony wzorzec zasad. Osiągnęli bardzo dobry poziom prawdopodobieństwa pojawienia się błędów w porównaniu do PCR. Niezawodność operacji ekstrakcji może być również zwiększona przez zastosowanie PCR do nie tylko jednej, lecz wielu probówek (badania Karpa, Kenyona i Waartsa). Boneh i Lipton zwiększyli odporność na błędy przez okresowe podwajanie liczby nici DNA w roztworze.

Uwagi końcowe.

Ostatnie lata badań nad DNA (pod kątem zastosowań znacznie różniących się od klasycznej medycyny czy inżynierii genetycznej) zaowocowały wieloma modelami obliczeniowymi o ciekawych własnościach i ich analizą, dokonywaną z wielu różnych perspektyw. Okazało się, że współpraca naukowców z różnych dziedzin i o różnych zainteresowaniach może być bardzo owocna w pracy nad zagadnieniami złożonymi. Przedstawiono tu jedynie zarys badań i zastosowań obliczeń opartych na DNA. Rezultaty znane obecnie wskazują, że komputery DNA nie zmienią w znaczący sposób podejścia i rozumienia systemów obliczeń efektywnych. Są jednak mocnym narzędziem do implementacji procesów równoległych.

Przyszłość maszyn molekularnych może być rozpatrywana przy założeniu „współpracy" z maszynami krzemowymi, które będą szybko wykonywać obliczenia sekwencyjne, zostawiając zagadnienia wymagające dużej równoległości komputerom DNA. Jest jeszcze za wcześnie, aby spodziewać się efektywnych realizacji takiego pomysłu, ale wciąż trwają badania biologów nad obniżeniem kosztu i zwiększeniem szybkości operacji molekularnych.

Sieci neuronowe.

Sieci neuronowe (SN) należą do dziedziny sztucznej inteligencji, może być traktowana jak pewna technika obliczeniowo- statystyczna. Matematycznie rzecz biorąc, dobrze skonstruowana SN jest w stanie "nauczyć się" aproksymować dowolną funkcję wielu zmiennych.

Zastosowania.

Do czego sieci się używa?

- do prognozowania (np. szeregów czasowych) – przewidywana jest wartość zmiennej na podstawie jej wartości wcześniejszych i ewentualnie innych, wpływających na wynik czynników

- do aproksymacji (znane są wartości funkcji w kilku punktach i należy znaleźć możliwe dobre przybliżenie wartości funkcji dla punktów leżących między nimi)
- do rozpoznawania kategorii - na podstawie znajomości charakterystyki pewnego zjawiska SN określa, do jakiej kategorii rozpatrywane zjawisko należy (m.in. rozpoznawanie pisma, cyfr, obrazów etc.)
- filtrowanie sygnałów
- kompresja obrazu, dźwięku
- inne zastosowania, np. sterowanie układami dynamicznymi.

We wszystkich tych zastosowaniach widać podaną wcześniej cechę podstawową: SN uczy się odwzorowywać funkcję, najczęściej zależną od wielu zmiennych. Ponieważ jest to aproksymacja, a nie interpolacja, to sieć jest w stanie **uogólniać** wiedzę, wykrywać istniejące pomiędzy zmiennymi zależności. W procesie uczenia sieć praktycznie rzecz biorąc tworzy statystyczny model uczonej zależności (funkcji).

Wadą sieci jest, po pierwsze, niemożność wyciągnięcia z nich, po nauczeniu, zależności między wejściem a wyjściem SN w jasnej postaci. Wiadomo, że sieć tworzy sobie model zjawiska, ale nie jest w pełni możliwe określenie tego modelu. Trudno ponadto dowieść ściśle, że dana (nauczona) sieć rzeczywiście potrafi sobie radzić z zadanym problemem. Te cechy SN stanowią wyzwanie dla teoretyków używających SN, natomiast praktykom wystarczy, ze sieć jest skuteczna (tym bardziej, że można sprawdzić, stosując odpowiednie metryki, na ile jest skuteczna).

Sieć jest zbudowana z "neuronów", ułożonych warstwami. W najprostszym modelu neurony tej samej warstwy nie są połączone między sobą, ale neurony dwóch sąsiednich warstw są połączone "każdy z każdym". Wyróżnia się warstwę wejściową (We), warstwy ukryte (U) (zazwyczaj 1 lub 2) i warstwę wyjściową (Wy). Sygnał przechodzi do warstwy We poprzez U aż do Wy, skąd jest odbierany i interpretowany. Sygnały podawane warstwie We muszą być numeryczne (liczbowe) i zazwyczaj wymaga się, by zawierały się w przedziale (-1, +1), (0, +1) albo jakimś podobnym, dlatego konieczne jest odpowiednie "zakodowanie" danych wejściowych sieci tak, by mogły one być przetworzone przez SN. Przygotowywanie danych wejściowych i projektowanie warstwy We jest zresztą zadaniem dość czasochłonnym

i wymagającym pewnej wprawy i doświadczenia (trzeba wiedzieć, w jakiej postaci sieć powinna dostać dane wejściowe, żeby nauczyć się szybciej i efektywniej).

Sygnał przechodząc przez dane połączenie wewnątrz SN zostaje pomnożony przez liczbę przypisaną temu połączeniu, zwaną wagą połączenia. Wagi mogą mieć dowolne wartości rzeczywiste.

Neurony każdej kolejnej warstwy (poza We) działają w następujący sposób:

1. zbierają wartości sygnałów dochodzących do nich z każdego z neuronów z warstwy poprzedniej
2. mnożą każdy z tych sygnałów przez odpowiednią wagę
3. sumują otrzymane wartości
4. suma ta jest następnie przeliczana na jakiś skończony przedział, np. (-1, +1); używa się do tego celu różnych funkcji matematycznych, najczęściej tzw. sigmoidy, ale też np. arcus tangensa albo funkcji schodkowej
5. uzyskana wartość staje się sygnałem tego neuronu i jest przekazywana do wszystkich neuronów z następnej warstwy.

Jak widać, korzystanie z nauczonej sieci jest dość proste i szybkie. Niestety, nie jest tak z samym uczeniem.

Uczenie to dobieranie wag połączeń między neuronami w taki sposób, aby po podaniu na wejście sieci jakichś wartości, na jej wyjściu uzyskać odpowiedni wynik. Oczywiście, nikt nie robi tego ręcznie, choć jest to możliwe. Stosuje się specjalne algorytmy. W przypadku sieci jednokierunkowych (w których sygnał jest rozprowadzany od warstwy We do Wy, w jednym tylko kierunku) uczy się je najczęściej w następujący sposób.

Przygotowuje się zestaw danych wejściowych wraz z odpowiadającymi im wynikami, jakie powinna uzyskać sieć. Wagi sieci inicjalizuje się w sposób przypadkowy. Następnie podaje się kolejno, wiele razy, wszystkie zestawy danych wejściowych i sprawdza, na ile odpowiedź sieci różni się od poprawnego wyniku. Oblicza się różnicę i następnie w odpowiedni sposób koryguje się wagi połączeń wewnątrz sieci (poczynając od połączeń od ostatniej warstwy ukrytej do warstwy wyjściowej, i tak dalej, aż do warstwy wejściowej).

Jeśli założymy wystarczającą cierpliwość „nauczyciela” i przygotujemy dane w odpowiedni sposób, to dostaniemy w wyniku sieć nauczoną, tj. mającą wartości wag takie,

jak trzeba, aby efektywnie rozwiązać postawione jej zadanie. Niestety, istotnym mankamentem SN jest fakt, iż uczenie zazwyczaj jest procesem bardzo czasochłonnym.

Nauczenie sieci nie jest ostatnim etapem jej projektowania – konieczne staje się jej przetestowanie - do tego wykorzystuje się kolejny zestaw danych wejściowych i wyjściowych, ale takich, które nie były wykorzystane podczas uczenia. Jeśli na tym zestawie sieć wypadnie pomyślnie, można się spodziewać, że w sytuacjach przyszłych (dla jakichś nieznanych danych) też tak będzie. Jeśli nie - trzeba powtórzyć uczenie, ewentualnie zmieniając budowę sieci albo zestaw danych uczących.

Nie jest ponadto trywialny sposób doboru danych stosowanych do uczenia i weryfikacji, stosuje się do tego odpowiednie metody statystyczne. Chodzi o to, żeby w czasie uczenia sieć miała styczność z danymi wejściowymi z wszystkich możliwych do przewidzenia kategorii, jakie mogą się pojawić później, na etapie stosowania już nauczonej sieci.

Uczenie nie jest też pierwszym stadium projektowania sieci. Najpierw trzeba ustalić, jak będzie ona zbudowana. Ilość neuronów wejściowych i wyjściowych jest określona z góry dla danego problemu, natomiast ilość neuronów w warstwach ukrytych (oraz ilość samych warstw, choć zazwyczaj stosuje się jedną, góra dwie) można regulować.

Generalnie rzecz biorąc, tylko jeden układ przestrzenny sieci jest optymalny. Jeśli warstwy ukryte mają zbyt wiele neuronów, sieć nadmiernie dopasowuje się do nauczanych danych (jej wynik staje się bliższy interpolacji) i przez to traci zdolność uogólniania. O takiej sieci mówi się, że jest "przeuczona". Jeśli natomiast neuronów w warstwach ukrytych jest za mało, sieć nie jest w stanie poprawnie odtworzyć uczonej funkcji, jest "niedouczona".

Istnieją pewne ogólne, empiryczne zasady doboru tych wartości i struktura każdorazowo powinna być optymalizowana. Można do tego stosować np. metody optymalizacji genetycznej.

Mając nauczoną sieć można ją przetestować i, jeśli sprawdziła się, stosować w praktyce.

Bibliografia.

1. Materiały na CD, DNA – computation.
2. L. Adleman „Molecular Computation of Solutions to Combinational Problems”, Science 266 (listopad, 1994)
3. T. Cormen, C. Leiserson, R. Rivest „Wprowadzenie do algorytmów”,WNT, Warszawa, 1997
4. A. Blum, T.Jiang, M. Li, J. Tromp, M.Yanakakis „Linear Approximation of Shortest Superstrings”, JACM (1994)
5. D. Beaver „Computing with DNA”, Journal of Computational Biology, 1995
6. D. Beaver, „Molecular Computing”: http://www.transarc.com/~beaver/research/alternative/molecule/molec.html
7. D. Boneh, R. Lipton, C. Dunworth „Breaking DES using a Molecular Computer”, http://www.cs.princeton.edu/~dabo
8. D. Rooss „Recent developments in DNA - computing”, Lehrstuhl fur Theoretische Informatik.
9. R. Poli, Introduction to Evolutionary Computation, http://www.cs.bham.ac.uk
10. R. Lipton, „DNA solution of Hard computational problems”, Science American, kwiecień, 1995
11. R. Lipton, „Using DNA to solve NP – complete problems”, http://www.cs.pronceton.edu/~dabo
12. D. K. Gifford, „On the path to computation with DNA”, Science American, listopad, 1994
13. http://www.neuron.phg.pl
14. http://umtii.fme.vutbr.cz/MECH/NN/tomgr1.html
15. http://www.rtis.com/nat/user/elsberry/zgists/wre/papers/ga_intro.html
16. http://www.nickbostrom.com/superintelligence.html
17. http://www.corninfo.chem.wisc.edu/writings/DNAoverview.html
18. http://citeseer.nj.nec.com/aharonov98quantum.html

19. http://citeseer.nj.nec.com/355223.html