Teoria grafów i analiza złożoności w świetle publikacji Graph Theory Aiman S Gannousa

AUTOR

Fundacja Dobre Państwo

STRESZCZENIE / ABSTRACT

Artykuł analizuje złożoność obliczeniową nie tylko jako domenę informatyki, ale jako miarę realnej wykonalności projektów w różnych dziedzinach. Autor podkreśla kluczowe rozróżnienie między złożonością pamięciową a czasową, wskazując na notację Big O jako narzędzie do oceny skalowalności systemów w obliczu rzeczywistych zbiorów danych. Tekst wprowadza również teorię grafów jako uniwersalny język opisu relacji, przekształcający abstrakcyjne powiązania między obiektami w ścisłe modele matematyczne. Od problemu mostów królewieckich po nowoczesne sieci, grafy pozwalają analizować strukturę kosztów i przepływów, zmieniając podejście z intuicyjnego na inżynieryjne. Całość stanowi refleksję nad przejściem od kultury deklaracji do kultury wykonalności w projektowaniu systemów.

The article analyzes computational complexity not only as a domain of computer science, but as a measure of the real feasibility of projects across various fields. The author emphasizes the key distinction between space and time complexity, pointing to Big O notation as a tool for assessing system scalability in the face of real-world datasets. The text also introduces graph theory as a universal language for describing relations, transforming abstract connections between objects into precise mathematical models. From the Seven Bridges of Königsberg problem to modern networks, graphs allow for the analysis of cost structures and flows, shifting the approach from intuitive to engineering-based. Overall, it is a reflection on the transition from a culture of declarations to a culture of feasibility in system design.

Artykuł przekształca techniczną teorię grafów i analizę złożoności obliczeniowej w rygorystyczne narzędzie do badania rzeczywistości społecznej, prawnej i ekonomicznej.

Autor stawia tezę, że wybór konkretnego algorytmu czy sposobu reprezentacji danych (np. macierzy sąsiedztwa vs list) nie jest jedynie decyzją techniczną, lecz aktem odpowiedzialności etycznej i politycznej, ponieważ determinuje on, co w systemie zostaje dostrzeżone, a co zignorowane. Poprzez analizę klasycznych problemów – od tras Eulera po dylemat P kontra NP – tekst dowodzi, że informatyka dostarcza języka do demaskowania ukrytych struktur władzy, wąskich gardeł w administracji oraz kosztów relacyjnych. W świecie, gdzie gospodarka i prawo opierają się na topologii powiązań, a nie tylko na zasobach, biegłość w teorii grafów staje się formą intelektualnej higieny, pozwalającą odróżnić realną wykonalność systemu od marketingowych deklaracji o jego nowoczesności.

SŁOWA KLUCZOWE

teoria grafów | analiza złożoności | notacja Big O | złożoność obliczeniowa | problemy NP-zupełne | P kontra NP | macierz sąsiedztwa | lista sąsiedztwa | algorytm Forda-Fulkersona | maksymalny przepływ | izomorfizm grafów | grafy dwudzielne | skalowalność algorytmiczna | sieci residualne | algorytm kwiatowy Edmondsa

Złożoność obliczeniowa jako miara realnej wykonalności

Złożoność obliczeniowa jest dyscypliną, która pozornie należy do informatyki, lecz w istocie opisuje znacznie więcej niż zachowanie programów. Opisuje granice działania: czasu, pamięci i organizacji. Granice państwa, rynku, przedsiębiorstwa, laboratorium, kancelarii, sieci logistycznej czy modelu sztucznej inteligencji. Kto rozumie złożoność, ten nie pyta naiwnie, czy coś „da się zrobić". Pyta dojrzale: jakim kosztem, w jakim czasie, przy jakim rozmiarze danych, ryzyku błędu, architekturze pamięci i odpowiedzialności decyzyjnej.

To przejście od kultury deklaracji do kultury wykonalności. W świecie algorytmów dobra wola bez analizy złożoności jest tylko elegancką formą niekompetencji. Podręcznik trafnie zaczyna od rozróżnienia złożoności pamięciowej i czasowej. Pierwsza dotyczy zasobów pamięci potrzebnych algorytmowi, druga tego, jak rośnie czas działania wraz ze wzrostem danych wejściowych. To rozróżnienie ma znaczenie cywilizacyjne. W epoce taniejących nośników łatwo ulec złudzeniu, że

pamięć przestała być problemem. Jest to jednak złudzenie księgowego patrzącego na magazyn, a nie inżyniera analizującego architekturę systemu. Pamięć bywa tańsza, ale nie jest darmowa – generuje koszty energetyczne, środowiskowe, infrastrukturalne, organizacyjne i prawne.

Model AI, baza danych, sieć społecznościowa czy system bankowy nie istnieją w próżni. Funkcjonują w centrach danych, budżetach, przepisach i ograniczeniach przepustowości. Dlatego złożoność pamięciowa nie znika; przestała jedynie być widoczna w prostych programach, by powrócić jako rachunek za skalę. Złożoność czasowa pozostaje jednak kluczowa, ponieważ czas jest najbardziej demokratycznym tyranem techniki.

Nie da się go dokupić po fakcie. Można zwiększyć liczbę serwerów, lecz nie można cofnąć opóźnienia decyzji, która miała zapaść w sekundę, a została obliczona po tygodniu. Dlatego analiza algorytmów bada nie tyle elegancję kodu, co tempo wzrostu kosztu. Jeden algorytm działa niemal tak samo szybko dla dziesięciu i miliona elementów. Drugi przy dziesięciu elementach wygląda jak geniusz, przy tysiącu jak urzędnik po urlopie, a przy milionie jak katastrofa budżetowa przebrana za procedurę. Właśnie tu pojawia się sens notacji Big O – języka asymptotycznego porządku. Nie chodzi o fetysz symbolu, lecz o chłodne pytanie: co stanie się z czasem działania, gdy świat przestanie być przykładem z tablicy, a stanie się rzeczywistym zbiorem danych. Współczesna nauka o algorytmach nie ocenia procedury po tym, że „działa” – to kryterium jest zbyt ubogie. Działać może także ręczne przeszukiwanie archiwum, jeśli dysponujemy tysiącem lat i cierpliwością mnicha. Kryterium naukowe brzmi: czy działa skalowalnie. Czy zachowuje rozsądną relację między rozmiarem wejścia a kosztem obliczeń. Czy potrafi przejść od eksperymentu do produkcji, od próbki do populacji, od makiety do infrastruktury.

Graf jako uniwersalny język opisu relacji i struktur

Z tego powodu podręcznikowa analiza algorytmów grafowych — od przeszukiwania wszerz i najkrótszych ścieżek, przez drzewa rozpinające, aż po przepływy i dopasowania — nie jest akademicką gimnastyką. Jest szkołą realizmu. Notatka źródłowa słusznie podkreśla, że operacje proste mogą mieć koszt stały, lecz pętle przechodzące przez wierzchołki i krawędzie ujawniają prawdziwą naturę problemu. Algorytm nie jest oceniany po pierwszym kroku, lecz po tym, jak zachowuje się podczas przejścia przez całą strukturę.

W tym miejscu zaczyna się teoria grafów. Jej wielkość polega na zdolności uczynienia relacji obiektem ścisłej analizy. Graf stanowi parę światów: wierzchołków i krawędzi. Wierzchołki reprezentują byty, a krawędzie powiązania — mogą to być miasta i drogi, osoby i znajomości, serwery i łącza, białka i interakcje, przepisy prawa i odesłania, firmy i transakcje, użytkownicy i

rekomendacje, dokumenty i cytowania czy modele językowe i wektory podobieństwa. Graf nie jest więc zwykłym rysunkiem, lecz antropologią relacji przetłumaczoną na matematykę dyskretną. Kto widzi tylko kropki i linie, widzi ilustrację; kto dostrzega wierzchołki i krawędzie, widzi strukturę władzy, kosztu, przepływu, zależności i ryzyka.

Historia grafów ma piękną właściwość: zaczyna się od problemu pozornie błahego. Siedem mostów królewieckich nie wyglądało na początek wielkiej teorii, lecz na miejską łamigłówkę. Pytanie brzmiało: czy można przejść przez każdy most dokładnie raz? Euler udowodnił, że istota sprawy nie tkwi w długości mostów ani w urodzie miasta, lecz w strukturze połączeń. To jeden z najważniejszych gestów nowoczesnej nauki: odrzucenie przypadkowego obrazu rzeczy na rzecz uchwycenia abstrakcyjnego mechanizmu. Przypowieść jest prosta: mieszczanin pyta, którędy iść; matematyk pyta, czy istnieje ścieżka; polityk pyta, czy można ogłosić sukces; inżynier zaś pyta, czy mosty mają właściwe stopnie. To ten ostatni zwykle ratuje budżet.

Termin „graf" pojawił się później, w kontekście chemii i reprezentacji struktur molekularnych. Jest to istotne, gdyż teoria grafów od początku nie była zamknięta w gabinecie matematyka, lecz służyła jako narzędzie opisu świata materialnego. Atom może być wierzchołkiem, a wiązanie krawędzią; podobnie serwer i kabel, użytkownik i interakcja czy firma i umowa. Graf stał się jednym z najważniejszych języków nowoczesności, ponieważ ta nie jest już światem rzeczy izolowanych, lecz światem powiązań. Gospodarka nie konkuruje już wyłącznie zasobami — Konkuruje topologią relacji. Państwo nie działa już tylko przez hierarchię, lecz przez sieci instytucjonalne. Prawo nie reguluje wyłącznie czynów, ale przepływy danych, odpowiedzialności, ryzyk i skutków.

Nawet kultura przestała być magazynem symboli, a stała się grafem cytowań, memów, autorytetów, konfliktów i zapożyczeń. Podstawowa terminologia grafów jest pozornie skromna, lecz jej siła rośnie z każdym zastosowaniem. Sąsiedztwo oznacza bezpośrednie połączenie dwóch wierzchołków. Incydencja wskazuje, że krawędź dotyka wierzchołka. Stopień wierzchołka określa liczbę związanych z nim krawędzi. Wierzchołek izolowany nie posiada połączeń, natomiast wierzchołek wiszący ma tylko jedno. Już te pojęcia pozwalają przełożyć matematykę na język socjologii i ekonomii.

Typologia grafów jako narzędzie analizy struktur władzy i kosztów

Wierzchołek izolowany to instytucja bez sieci kontaktów, natomiast wierzchołek wiszący to podmiot zależny od jednego kanału dostępu. Wierzchołek o wysokim stopniu staje się centrum

relacyjnym – potencjalnym hubem, brokerem informacji czy punktem kontroli, ale jednocześnie punktem awarii. Gdy w sieci wszystko przechodzi przez jeden centralny węzeł, efektywność bywa wysoka, lecz odporność drastycznie spada. To stara prawda organizacyjna ubrana w precyzyjną postać grafową: centralizacja daje szybkość, ale prosi się o kryzys, jeśli zabraknie redundancji.

Twierdzenie o uściskach dłoni, według którego suma stopni wszystkich wierzchołków w grafie nieskierowanym jest równa podwojonej liczbie krawędzi, brzmi jak matematyczna anegdota, a w rzeczywistości stanowi lekcję rachunkowości relacji. Każda relacja ma dwa końce; każde połączenie obciąża strukturę po obu stronach. W świecie społecznym oznaczałoby to, że nie istnieje więź jednostronnie bezkosztowa. W gospodarce – że każda transakcja generuje ślad po dwóch stronach. W prawie zaś każda relacja tworzy potencjalne uprawnienie i obowiązek. Graf uczy, że sieć nie jest mgłą, lecz bytem policzalnym. A to, co policzalne, można zaprojektować, zoptymalizować, skontrolować lub zepsuć z imponującą precyzją.

Klasyfikacja grafów pozwala dobrać model do konkretnego problemu. Graf prosty nie dopuszcza pętli ani wielokrotnych krawędzi – jest elegancki, lecz często zbyt ascetyczny dla rzeczywistości. Multigraf, dopuszczając wiele krawędzi między tymi samymi wierzchołkami, lepiej opisuje systemy redundantne: na przykład kilka niezależnych połączeń między serwerami lub alternatywne kanały prawne między podmiotami. Pseudograf idzie dalej, dopuszczając pętle, a więc relacje obiektu z samym sobą.

Graf skierowany wprowadza orientację: relacja od A do B nie musi oznaczać relacji od B do A. Przelew ma kierunek. Polecenie ma kierunek. Cytowanie i zależność technologiczna również mają kierunek. Władza bardzo często udaje symetrię, ale graf skierowany zdejmuje z niej kostium balowy.

Graf ważony dodaje do krawędzi koszt, dystans, przepustowość, ryzyko lub wartość. To przejście od mapy relacji do ekonomii relacji. Dwie drogi mogą łączyć te same miasta, ale różnić się kosztem; dwa kanały komunikacji mogą łączyć instytucje, lecz różnić się niezawodnością; dwie procedury prawne mogą prowadzić do tego samego rezultatu, mając jednak odmienną czasochłonność i ryzyko procesowe. Graf ważony jest zatem modelem dojrzałego świata, który nie pyta wyłącznie o istnienie połączenia, lecz o to, ile ono kosztuje, jak długo trwa, jakie niesie ryzyko i kto ponosi konsekwencje jego użycia.

Właściwości grafów prowadzą nas od samego faktu istnienia połączeń do pytania o możliwość nawigacji. Spójność oznacza, że między każdą parą wierzchołków istnieje ścieżka. W świecie technicznym to warunek komunikacji, społecznym – integracji, a prawnym – skuteczności instytucjonalnej. System może posiadać wiele podmiotów, aktów i kanałów, a mimo to pozostać

niespójny. Przypomina wtedy archipelag procedur, w którym każda wyspa ma własny regulamin, ale nikt nie zbudował promu. Grafowa kategoria spójności pozwala precyzyjnie nazwać to, co w administracji zbyt często ukrywa się pod mglistym słowem „koordynacja".

Graf skierowany komplikuje sprawę, wprowadzając spójność silną i słabą. Silna oznacza, że można dotrzeć z każdego wierzchołka do każdego innego z poszanowaniem kierunków krawędzi. Słaba zaś sugeruje, że po zignorowaniu kierunków struktura wydaje się połączona. To znakomita metafora wielu systemów instytucjonalnych: na papierze wszyscy są połączeni, lecz w praktyce informacja płynie tylko w jedną stronę. Obywatel może wysłać wniosek, ale nie otrzyma odpowiedzi; pracownik raportuje do góry, lecz decyzja nie wraca z uzasadnieniem; dane zbierane centralnie nie wracają jako użyteczna wiedza lokalna. To nie jest silna spójność – to słaba spójność z propagandą sukcesu.

P kontra NP jako granica między weryfikacją a odkryciem

W tym krajobrazie pojawiają się grafy eulerowskie i hamiltonowskie. Różnica między nimi jest dydaktycznie bezcenna. Graf eulerowski skupia się na krawędziach – celem jest przejście przez każdą z nich dokładnie raz. Graf hamiltonowski koncentruje się na wierzchołkach; dąży do odwiedzenia każdego z nich dokładnie raz. Brzmi to podobnie, ale obliczeniowo to zupełnie inny świat.

Dla grafów eulerowskich istnieją eleganckie kryteria oparte na stopniach wierzchołków. W przypadku hamiltonowskich sprawa staje się dramatycznie trudniejsza. To jedna z tych różnic, które uczą pokory wobec pozornych analogii. Nie wszystko, co brzmi podobnie w języku naturalnym, zachowuje się tak samo w matematyce. Algorytm nie ulega retoryce. Algorytm jest bezlitosnym korektorem naszych intuicji. Tu pojawia się pierwsza istotna aktualizacja i zarazem życzliwa korekta notatki.

Klasa P obejmuje problemy decyzyjne rozwiązywalne w czasie wielomianowym. Klasa NP zawiera problemy, dla których zaproponowane rozwiązanie można zweryfikować w czasie wielomianowym. Nie należy twierdzić, że NP to po prostu problemy „nierozwiązane optymalnie" lub że z definicji cechują się złożonością wykładniczą. Istota pytania P kontra NP brzmi: czy to, co da się szybko sprawdzić, da się także szybko znaleźć? Clay Mathematics Institute ujmuje tę intuicję właśnie przez różnicę między łatwością weryfikacji a trudnością znalezienia rozwiązania; zagadnienie to pozostaje jednym z problemów milenijnych.

To rozróżnienie ma ogromną wagę poza informatyką. W prawie łatwiej sprawdzić, czy dokument spełnia wymogi, niż zaprojektować optymalną umowę. W ekonomii łatwiej ocenić skuteczność danego przydziału zasobów, niż odnaleźć ten najlepszy w ogromnej przestrzeni możliwości. W zarządzaniu łatwiej stwierdzić wykonalność harmonogramu, niż stworzyć ten idealny. W bioinformatyce szybciej ocenimy zgodność danych z hipotezą, niż odkryjemy właściwą strukturę od podstaw. W kulturze łatwiej rozpoznać trafną interpretację, niż wyprodukować ją mechanicznie.

P kontra NP nie jest więc jedynie pytaniem o maszyny Turinga. To pytanie o różnicę między kontrolą a twórczością, audytem a odkryciem, certyfikatem a wynalazkiem. Problemy NP-zupełne stanowią najtwardsze jądro klasy NP. Należą do niej i mają tę właściwość, że każdy inny problem z NP można do nich sprowadzić w czasie wielomianowym. To nie jest sucha definicja – oznacza to, że jeden przełom mógłby wywołać lawinę.

Gdyby znaleziono wielomianowy algorytm dla dowolnego problemu NP-zupełnego, uzyskano by wielomianowe rozwiązania dla wszystkich problemów z NP. Wówczas wiele fundamentów kryptografii, optymalizacji i automatycznego dowodzenia musiałoby zostać przewartościowanych. Nie oznacza to, że „jutro rano upadnie świat” – praktyka zależy od stałych, rozmiarów i implementacji – ale fundament pojęciowy zostałby przesunięty. Matematyka rzadko wywołuje trzęsienia ziemi, lecz P kontra NP jest jednym z tych miejsc, gdzie sejsmograf stoi od dawna pod napięciem.

Problemy NP-trudne są jeszcze subtelniejsze. Są co najmniej tak trudne jak problemy NP-zupełne, ale nie muszą same należeć do klasy NP. Dlatego należy ostrożnie operować pojęciem weryfikacji. Optymalizacyjna wersja problemu komiwojażera, w której szukamy najkrótszej możliwej trasy, jest klasycznym przykładem problemu NP-trudnego. Natomiast wersja decyzyjna – pytająca, czy istnieje trasa o koszcie nie większym niż określona wartość – należy do NP i jest NP-zupełna. W języku biznesu różnica ta jest wyraźna: czym innym jest sprawdzenie, czy oferta mieści się w budżecie, a czym innym znalezienie najlepszej z astronomicznej liczby kombinacji. Pierwsze to kontrola progu, drugie to eksploracja przestrzeni.

Osobnej uwagi wymaga izomorfizm grafów. Notatka źródłowa słusznie wskazuje, że chodzi o rozpoznanie, czy dwa grafy mają tę samą strukturę mimo odmiennego nazewnictwa wierzchołków. Nie należy jednak traktować go jako typowego problemu NP-zupełnego. Jego status jest bardziej wyrafinowany: należy do NP, ale nie wiadomo, czy jest w P, i nie uznano go za NP-zupełny. Co więcej, praca László Babai'ego wprowadziła algorytm quasi wielomianowy, co znacząco zmieniło rozumienie jego pozycji w pejzażu złożoności. To dowód na to, że mapa trudności obliczeniowej nie jest czarno-biała; istnieją problemy żyjące w strefach przejściowych, oporne na proste etykiety.

Dygresja kulturoznawcza jest tu nieunikniona. Człowiek kocha etykiety, bo etykiety dają odpoczynek od myślenia. „Łatwe", „trudne", „rozwiązane", „nierozwiązane", „praktyczne", „teoretyczne". Teoria złożoności niszczy tę wygodę. Mówi wprost: łatwość zależy od modelu obliczeń, rozmiaru wejścia, reprezentacji danych, struktury instancji i wymaganego poziomu pewności.

Reprezentacja grafu jako akt decyzji i odpowiedzialności

To jest anty populizm matematyczny. Nie pozwala sprzedać skomplikowanego świata w trzech hasłach i jednym slajdzie, dlatego jest tak cenna dla profesjonalistów – uczy intelektualnej higieny. W epoce sztucznej inteligencji, automatyzacji i modelowania sieciowego jest to higiena pierwszej potrzeby. Z tej perspektywy teoria grafów staje się pomostem między matematyką a instytucjonalną odpowiedzialnością.

Gdy projektujemy bazę danych, wybieramy nie tylko strukturę informatyczną, ale model poznawczy organizacji. Projektując sieć komputerową, decydujemy nie tylko o topologii kabli, lecz o dopuszczalnej architekturze awarii. Tworząc algorytm dopasowania, wybieramy nie tylko technikę optymalizacji, ale regułę sprawiedliwego lub efektywnego przydziału. Kiedy zaś projektujemy system AI oparty na grafach wiedzy, wektorach podobieństwa i relacjach między dokumentami, wybieramy nie tylko sposób wyszukiwania, lecz epistemologię maszyny.

Graf jest więc narzędziem, ale nigdy nie jest narzędziem niewinnym. Każda krawędź to decyzja o tym, co uznajemy za relację. Każda waga to decyzja o tym, co uznajemy za koszt. Każdy brak krawędzi to decyzja o tym, czego system nie widzi. Graf, zanim stanie się algorytmem, musi zostać zapisany. To zdanie brzmi niewinnie, lecz zawiera jedną z najważniejszych prawd informatyki stosowanej: nie istnieje czyste obliczanie bez wcześniejszej decyzji o reprezentacji.

Komputer nie widzi grafu jako kółek i linii. Nie dostrzega mapy, sieci, relacji społecznej, zależności prawnej ani powiązań białek w organizmie – widzi strukturę danych. To, co człowiek rozpoznaje intuicyjnie jako układ połączeń, maszyna musi otrzymać w formie tablicy, macierzy, listy, wskaźnika, rekordu, indeksu, obiektu lub własności. Reprezentacja grafu jest zatem aktem translacji świata relacji na język pamięci.

Książka słusznie wyróżnia trzy klasyczne sposoby reprezentowania grafów: macierze sąsiedztwa, macierze incydencji oraz listy sąsiedztwa. Każda z nich odpowiada na inne pytanie praktyczne. Macierz sąsiedztwa pyta: czy między dwoma wierzchołkami istnieje krawędź? Macierz incydencji

pyta: które krawędzie dotykają których wierzchołków? Lista sąsiedztwa zaś pyta: z kim bezpośrednio połączony jest dany wierzchołek?

Te trzy pytania tworzą trzy odmienne filozofie dostępu do relacji. W pierwszej dominuje błyskawiczna kontrola konkretnego połączenia, w drugiej formalna ewidencja udziału w relacjach, a w trzeciej ekonomiczna nawigacja po sąsiedztwie. Informatyka, podobnie jak administracja, nie znosi abstrakcyjnych deklaracji o „dostępności danych". Pyta brutalnie: dostępności do czego, w jakim czasie i jakim kosztem?

Macierz sąsiedztwa jest rozwiązaniem eleganckim. Dla grafu o określonej liczbie wierzchołków tworzy się dwuwymiarową strukturę, w której przecięcie wiersza i kolumny informuje o istnieniu krawędzi między odpowiadającymi im wierzchołkami. W grafie nieskierowanym taka macierz jest symetryczna, gdyż relacja między A i B jest równoważna relacji między B i A. W grafie skierowanym symetria znika – krawędź od A do B nie oznacza automatycznie krawędzi od B do A. Ta różnica nie jest wyłącznie techniczna; uczy ona, że wzajemność nie jest domniemaniem matematycznym. Trzeba ją zapisać. W stosunkach społecznych, gospodarczych i prawnych jest podobnie: zależność, obowiązek, przepływ informacji i odpowiedzialność mają swój kierunek, nawet gdy retoryka próbuje je wygładzić.

Wybór reprezentacji danych jako architektura myślenia i odpowiedzialności

Zaletą macierzy sąsiedztwa jest natychmiastowa odpowiedź na pytanie, czy dwa wierzchołki są połączone. To jak posiadanie centralnego rejestru, w którym jedno spojrzenie wystarcza, by potwierdzić istnienie relacji. Wadą pozostaje koszt pamięciowy, szczególnie w grafach rzadkich. Jeśli mamy milion potencjalnych podmiotów i stosunkowo niewiele realnych połączeń, macierz wypełnia się ogromną przestrzenią zer. To biurokratyczny sen o idealnym formularzu, który przewiduje każdą możliwość, ale pozostaje głównie pusty. W systemach rzeczywistych ta pustka również kosztuje: pamięć, energię, czas przesyłu i złożoność utrzymania, a niekiedy cierpliwość użytkownika – zasobu często bardziej deficytowego niż budżet na serwery.

Lista sąsiedztwa podchodzi do tego problemu z większą pokorą wobec rzeczywistości. Zamiast tworzyć pełną tablicę wszystkich możliwych relacji, zapisuje dla każdego wierzchołka jedynie jego faktycznych sąsiadów. Jest to rozwiązanie naturalne dla grafów rzadkich, a więc dla ogromnej części systemów praktycznych. Sieć dróg nie łączy każdego miasta z każdym innym bezpośrednio; sieć społeczna nie wiąże każdej osoby z każdą inną; sieć cytowań czy prawna również nie łączą

wszystkich elementów w sposób totalny. Lista sąsiedztwa zachowuje zatem ekonomię świata, zamiast udawać jego pełną gęstość. Jej słabością jest konieczność przejrzenia listy sąsiadów, by sprawdzić konkretną krawędź. Jeśli kluczowe jest pytanie: „czy A jest połączone z B?", macierz może być lepsza. Jeśli jednak pytamy: „dokąd można pójść z A?", lista okazuje się bardziej rozsądna.

Macierz incydencji ma odmienny temperament. Nie opisuje relacji przez pary wierzchołków, lecz przez związek wierzchołków z krawędziami. W grafach nieskierowanych pozwala wskazać, które krawędzie dotykają danego wierzchołka; w skierowanych może wykorzystywać znaki wskazujące wejście i wyjście krawędzi. Reprezentacja ta ma szczególne znaczenie tam, gdzie sama krawędź staje się istotnym obiektem analizy. W sieciach przepływu krawędź nie jest tylko kreską między punktami – jest kanałem, nośnikiem przepustowości, ryzyka, kosztu lub napięcia.

W prawie krawędź może być umową, pełnomocnictwem, relacją zależności, cesją, zobowiązaniem albo delegacją kompetencji. W ekonomii może być transakcją, w energetyce linią przesyłową, a w bioinformatyce interakcją biologiczną. Macierz incydencji mówi zatem: nie patrz tylko na aktorów, patrz także na same więzi, bo więzi mają własną ontologię. Wybór reprezentacji grafu jest pierwszą wielką lekcją teorii złożoności w praktyce. Ten sam problem może być obliczeniowo rozsądny albo absurdalny, zależnie od sposobu zapisu danych. Jest to szczególnie ważne dla organizacji mylących cyfryzację z digitalizacją dokumentów. Można przenieść chaos papierowy do systemu elektronicznego i nadal mieć chaos, tyle że szybciej indeksowany. Reprezentacja danych jest architekturą myślenia instytucji. Jeśli relacje są przechowywane tak, że każde pytanie wymaga kosztownych złączeń, obejść i interpretacji, system będzie „nowoczesny" jedynie w sensie marketingowym. W sensie obliczeniowym pozostanie labiryntem z lampkami LED.

Stąd przejście do baz danych grafowych jest naturalne. Klasyczne bazy relacyjne są wielkim osiągnięciem informatyki i nie należy ich pochopnie wysyłać na emeryturę – są znakomite tam, gdzie dane mają stabilne schematy, jasne encje i przewidywalne zapytania. Jednak w świecie głęboko połączonych danych, gdzie istotą problemu jest relacja i jej wielokrotne przechodzenie, model grafowy bywa poznawczo i wydajnościowo bardziej adekwatny. Neo4j opisuje grafową bazę danych jako sposób przechowywania informacji w postaci węzłów, relacji i właściwości, a nie w klasycznych tabelach czy dokumentach.

Ta różnica nie jest kosmetyczna. Oznacza ona, że relacja staje się obywatelem pierwszej kategorii w modelu danych, a nie skutkiem ubocznym złączenia tabel. W tradycyjnej bazie relacyjnej pytanie o dalekie powiązania często prowadzi do serii złączeń. Każde z nich jest rodzajem administracyjnego mostu między tabelami. Przy jednym lub dwóch przejściach mosty są znośne, lecz przy wielu poziomach powiązań zaczynają przypominać wyprawę przez archiwum, w którym każdy segregator

odsyła do kolejnego pokoju, a każdy pokój wymaga osobnej przepustki. Grafowa baza danych próbuje odwrócić tę logikę.

Graf jako aparat demaskujący ukryte struktury

Gdy węzły bezpośrednio wskazują na swoje relacje i sąsiadów, nawigacja po grafie staje się naturalnym sposobem zadawania pytań o to, kto jest powiązany z kim, przez jakie kanały, na jakiej głębokości, jakim kosztem i za pośrednictwem kogo. Neo4j określa to podejście jako natywną pracę z grafem i analizę relacji, a jego biblioteka Graph Data Science dostarcza algorytmów grafowych, technik uczenia maszynowego oraz procedur analitycznych. W tym punkcie grafowa baza danych przestaje być wyłącznie narzędziem informatyka, stając się instrumentem ekonomisty, prawnika czy kulturoznawcy. Ekonomista dostrzeże w niej sieci dostaw, zależności kapitałowe, ryzyko koncentracji, powiązania właścicielskie, przepływy wartości i podatność systemu na szoki. Prawnik zidentyfikuje relacje zobowiązań, strukturę grup kapitałowych, łańcuchy odpowiedzialności, konflikty interesów, powiązania beneficjentów rzeczywistych oraz mapę ryzyka regulacyjnego. Kulturoznawca natomiast zobaczy sieci wpływu, przepływ symboli, cytowań, mód i autorytetów, a także konflikty narracyjne.

W każdym z tych przypadków graf jest aparatem demaskującym. Ujawnia to, co pozostaje niewidoczne z perspektywy pojedynczego rekordu. Podczas gdy rekord mówi: „oto podmiot”, graf pyta: kto go niesie, kto finansuje, kto cytuje, kto kontroluje, kto od niego zależy i kto udaje, że nie ma z nim nic wspólnego. Bazy grafowe są szczególnie użyteczne tam, gdzie pytania przyjmują formę ścieżek. W systemach rekomendacyjnych badamy, jak użytkownik, produkt, kategoria, zakup, ocena i podobieństwo tworzą wzorzec decyzji. W systemach przeciwdziałania nadużyciom sprawdzamy, czy pozornie odrębne konta, adresy, urządzenia, transakcje i pośrednicy nie tworzą podejrzanej struktury.

W systemie naukowym analizujemy, jak publikacje cytują się wzajemnie, które idee stanowią mosty między dziedzinami, które artykuły są hubami, a które jedynie głośno brzęczą na peryferiach. W systemie administracyjnym pytamy, czy procedura rzeczywiście prowadzi obywatela do rozstrzygnięcia, czy raczej do rytualnego krążenia między węzłami instytucjonalnymi. Graf jest bezlitosny wobec labiryntów udających procedury.

Należy jednak unikać naiwnego kultu technologii; grafowa baza danych nie jest magicznym eliksirem. Błędna definicja węzłów, niewłaściwe nazewnictwo relacji, pominięcie istotnych wag, zignorowanie jakości danych czy pomylenie korelacji z przyczynowością sprawią, że graf jedynie szybciej doprowadzi nas do błędnych wniosków. Najgroźniejszy nie jest zatem brak grafu, lecz graf

pozornie precyzyjny. Sieć może sprawiać wrażenie dowodu, będąc w rzeczywistości jedynie estetycznie uporządkowanym zbiorem założeń. Profesjonalista musi więc pytać: co oznacza krawędź, kto ją ustalił, jak została zweryfikowana, czy jest symetryczna, skierowana, czy posiada wagę, czy jest aktualna i czy wynika z obserwacji, czy z interpretacji. Graf bez epistemologii jest infografiką z ambicjami prokuratora.

Reprezentacja grafów ma także bezpośrednie znaczenie dla sieci komputerowych. Topologia gwiazdy, pierścienia czy struktury hybrydowe to nie tylko szkolne schematy, lecz odmienne koncepcje odporności, kosztu i kontroli.

Graf jako język infrastruktury i systemów krytycznych

Topologia gwiazdy opiera się na centralnym węźle; jest łatwa w zarządzaniu, lecz centrum staje się punktem krytycznym. Pierścień rozkłada połączenia w strukturze cyklicznej, co nadaje inną dynamikę przesyłu i awarii, natomiast hybryda próbuje łączyć zalety obu rozwiązań. Multigraf pozwala z kolei modelować połączenia redundantne, czyli kilka krawędzi między tymi samymi węzłami. Redundancja jest tu słowem kluczowym. W gospodarce cyfrowej jedno połączenie to prośba o awarię. Dwa połączenia to początek odpowiedzialności. Trzy połączenia to nie przesada, lecz kultura ciągłości działania.

Sieci komputerowe dowodzą, że graf jest językiem infrastruktury. Pakiet danych nie podróżuje przez metaforę, lecz przez konkretną trasę. Routing jest praktyką grafową: przepustowość staje się wagą lub ograniczeniem krawędzi, opóźnienie kosztem, a awaria węzła czy łącza – usunięciem wierzchołka bądź krawędzi. Atak na system może być próbą przeciążenia wybranych węzłów, przejęcia mostów między składowymi lub zakłócenia centralnych hubów.

Z tego powodu wiedza grafowa jest niezbędna dla cyberbezpieczeństwa. Nie wystarczy świadomość, że system składa się z wielu elementów; trzeba wiedzieć, które z nich są artykulacjami, które krawędzie pełnią funkcję mostów, gdzie istnieją ścieżki alternatywne i które obszary są silnie spójne, a które jedynie udają spójność, gdy zignorujemy kierunek przepływu. W inżynierii oprogramowania grafy pojawiają się równie naturalnie: kod ma zależności, moduły importują moduły, funkcje wywołują funkcje, a testy pokrywają ścieżki wykonania. Maszyny stanów opisują możliwe przejścia systemu, zaś kompilatory analizują grafy przepływu sterowania i zależności danych.

Architektura mikroserwisowa jest grafem komunikacji, który może być elegancki albo potworny. W wersji eleganckiej relacje są czytelne, odpowiedzialności rozdzielone, a ścieżki awaryjne znane. W

wersji potwornej każdy mikroserwis rozmawia z każdym bez wyraźnego powodu, a diagram architektury wygląda jak makaron po burzy. Graf nie naprawi takiego systemu automatycznie, ale pozwala przestać udawać, że problem jest „komunikacyjny" – on jest topologiczny. To prowadzi bezpośrednio do kwestii testowania i weryfikacji, gdzie grafy umożliwiają modelowanie ścieżek wykonania oraz stanów aplikacji.

W systemach krytycznych – takich jak urządzenia medyczne, infrastruktura energetyczna, finanse czy transport – nie wolno polegać na intuicji, że „większość przypadków działa". Profesjonalizm zaczyna się tam, gdzie projektant pyta, jakie ścieżki są możliwe i osiągalne, które prowadzą do stanu błędnego, które wymagają testów regresyjnych, a które stanowią rzadki, lecz groźny wariant. Graf ścieżek wykonania jest tu prawnikiem systemu: żąda wykazania, że procedura nie tylko ma dobrą intencję, ale także nie produkuje katastrofy przy brzegowych warunkach.

W bioinformatyce grafy zyskują nową głębię, gdyż organizmy są systemami relacji. Białka oddziałują z białkami, a geny uczestniczą w sieciach regulacyjnych. Sekwencje DNA można składać z fragmentów przy użyciu grafów reprezentujących nakładanie się odczytów lub struktur de Bruijna. Choroby nie są tu punktowymi zdarzeniami, lecz zaburzeniami sieci oddziaływań. Farmakologia coraz częściej analizuje nie pojedynczy cel molekularny, lecz jego pozycję w sieci biologicznej.

Współczesne opracowania dotyczące grafowych sieci neuronowych (GNN) w bioinformatyce pokazują ich zastosowanie w przewidywaniu związków genów z chorobami, odkrywaniu leków, analizie funkcji białek, integracji danych multiomicznych i budowie biomedycznych grafów wiedzy. To moment, w którym klasyczna teoria grafów spotyka sztuczną inteligencję. GNN próbują uczyć się reprezentacji węzłów, krawędzi i całych struktur, uwzględniając sąsiedztwo. Nie zastępują one klasycznych algorytmów, lecz uzupełniają arsenał metod w problemach predykcyjnych o wysokiej złożoności danych. W przemyśle GNN modelują zależności relacyjne w wielu domenach – od rekomendacji i wykrywania nadużyć po systemy wiedzy.

Grafowe sieci neuronowe nie anulują jednak problemów klasycznej teorii grafów; one je dziedziczą i komplikują. Jeśli graf wejściowy jest źle skonstruowany, model nauczy się eleganckiej wersji błędu. Jeśli relacje są historycznie stronnicze, model może utrwalić nierówności. Gdy wagi krawędzi wynikają z niejawnych decyzji instytucjonalnych, algorytm nadaje im pozór obiektywności. Z kolei system rekomendacyjny wzmacniający popularne węzły może produkować efekt „bogacenia się bogatych", gdzie huby stają się jeszcze silniejszymi centrami, a peryferie zostają jeszcze bardziej zmarginalizowane.

Grafy jako uniwersalny język relacji społecznych i systemowych

To jest moment, w którym ekonomista, prawnik i kulturoznawca muszą wejść do serwerowni. Nie po to, by przeszkadzać inżynierom, lecz by przypomnieć, że dane relacyjne są w istocie danymi społecznymi. Przypowieść o trzech kartografach dobrze oddaje sens reprezentacji grafów. Pierwszy narysował mapę wszystkich możliwych dróg między wszystkimi miastami – była kompletna i imponująca, ale tak wielka, że nie dało się jej rozłożyć. Drugi zapisał tylko drogi istniejące; jego mapa była praktyczna, choć odnalezienie konkretnej trasy wymagało czasem wertowania list. Trzeci opisał każdą drogę jako osobny byt, uwzględniając mosty, koszty, ograniczenia i kierunki.

Dopiero razem pokazali, że nie istnieje jedna reprezentacja doskonała. Istnieje jedynie ta adekwatna do pytania, a to właśnie pytanie jest królem architektury danych. Z punktu widzenia prawa reprezentacja grafowa rodzi szczególne wyzwania. Relacje między osobami, firmami i zdarzeniami mogą ujawniać więcej niż pojedyncze rekordy. Czasem pojedyncza informacja pozostaje niewinna, lecz jej miejsce w grafie staje się wrażliwe. Ktoś może nie ujawniać jawnie swoich preferencji czy sieci kontaktów, a jednak analiza grafowa potrafi to wydedukować poprzez sąsiedztwo, wspólnotę połączeń lub powtarzalne ścieżki. Prawo ochrony danych osobowych, prawo konkurencji, compliance czy regulacje AI muszą traktować dane nie tylko jako rekordy, ale jako relacje. Wiek tabel uczył ochrony pól. Wiek grafów wymaga ochrony powiązań.

W ekonomii grafy są narzędziem rozumienia ukrytej struktury kosztów. Łańcuch dostaw to nie lista kontrahentów, lecz graf zależności. Rynek pracy nie jest zbiorem CV i ofert, lecz grafem kompetencji, instytucji, reputacji i przepływów informacji. System finansowy nie jest sumą bilansów, lecz grafem ekspozycji. Miasto natomiast nie jest zbiorem adresów, lecz grafem dojazdów, usług i nierówności przestrzennych. Dlatego decyzje publiczne i korporacyjne podejmowane bez myślenia grafowego są często ślepe na skutki drugiego i trzeciego rzędu. Widzą węzeł, nie widzą sieci. Widzą koszt lokalny, nie widzą przesunięcia obciążenia. Widzą oszczędność, nie widzą utraty odporności.

Z perspektywy kulturoznawstwa grafy pozwalają analizować obieg znaczeń bez popadania w sentymentalną mgłę. Idee mają swoje sąsiedztwa, a narracje – huby. Autorytety tworzą mosty między wspólnotami interpretacyjnymi, zaś konflikty memetyczne bywają starciami o centralność w grafie uwagi. Platformy cyfrowe nie tylko rozpowszechniają treści, ale konstruują topologie widzialności: to, co jest blisko w grafie rekomendacji, staje się bliskie w świadomości użytkownika. To, co zostaje odcięte, znika jak wierzchołek izolowany. Kultura algorytmiczna nie cenzuruje wyłącznie zakazem – czasem robi to poprzez dystans.

Dochodzimy tu do współczesnego paradygmatu naukowego: świat złożony należy badać jako sieć relacji, a nie magazyn atomów informacyjnych. Nie oznacza to naiwnego przekonania, że wszystko jest grafem, lecz że stał się on jednym z najbardziej produktywnych modeli formalnych tam, gdzie kluczowa jest struktura powiązań. Od baz danych i bioinformatyki po AI i nauki społeczne – graf dostarcza wspólnego języka. Jego siła polega na tym, że pozwala jednocześnie modelować i obliczać; nie tylko opowiada o powiązaniach świata, ale pozwala je precyzyjnie policzyć.

Każdy paradygmat jednak potrzebuje krytyków, by chronić się przed pychą. Część z nich wskazuje, że redukcja zjawisk społecznych do węzłów i krawędzi może spłaszczać znaczenie relacji – nie każda więź ma bowiem ten sam status ontologiczny. Inni ostrzegają przed fetyszyzacją centralności: to, co centralne, nie zawsze jest najważniejsze moralnie czy poznawczo. Jeszcze inni zauważają, że modele grafowe w AI mogą wzmacniać nierówności, ucząc się z danych historycznych, a historia, jak wiadomo, nie była neutralnym laborantem w białym fartuchu. Te głosy nie unieważniają grafów, lecz wymagają od nas epistemicznej dyscypliny zamiast nabożeństwa.

Dygresja praktyczna: organizacja chcąca korzystać z grafów musi najpierw nauczyć się zadawać pytania grafowe. Nie „jakie mamy dane?”, lecz „jakie mamy relacje?”. Nie „ile mamy rekordów?”, lecz „jaką mamy topologię?”. Nie „kto jest w bazie?”, lecz „kto z kim, przez co i z jakim skutkiem jest powiązany?”. Zamiast pytać, czy system działa, należy zapytać: które węzły są krytyczne, gdzie są mosty, a gdzie centrum udające całość?

To jest różnica między informatyzacją a inteligencją organizacyjną. Algorytm grafowy pozwala zadać pytanie tak precyzyjne, by odpowiedź wygenerowała procedura, a nie intuicja. Graf bez algorytmu to elegancka abstrakcja i mapa relacji; algorytm wprawia tę strukturę w ruch. Pyta o optymalne trasy, największe przepływy i krytyczne krawędzie – te, które nie są jedynie kosztownymi marmurami w gmachu administracji niepotrafiącej odpowiedzieć na pismo w terminie. Algorytm jest więc nie tylko metodą obliczeń, ale dyscypliną odpowiedzialności. Wymusza przejście od „mniej więcej wiadomo” do konkretnego modelu, określonych danych i czasu.

Kanon algorytmiki grafowej jako fundament analizy

Podręcznik słusznie przeprowadza czytelnika przez fundamenty algorytmiki grafowej: od tras Eulera i algorytmów Fleury'ego oraz Hierholzera, przez najkrótsze ścieżki Dijkstry i Bellmana-Forda, aż po minimalne drzewa rozpinające Prima i Kruskala. Katalog ten uzupełniają maksymalne przepływy Forda-Fulkersona i Dinica, problemy dopasowania rozwiązywane przez Hopcrofta-Karpa oraz algorytm kwiatowy Edmondsa. Ten zestaw nie jest przypadkową listą nazwisk.

Algorytmy jako lekcje stylu myślenia i zarządzania

To genealogia nowoczesnego zarządzania relacjami. Każdy z tych algorytmów odpowiada na inny typ problemu praktycznego, a zarazem uczy innego stylu myślenia. Euler pokazuje, kiedy możliwa jest trasa przez wszystkie krawędzie. Dijkstra uczy ekonomii najkrótszej drogi. Bellman-Ford ostrzega przed ujemnymi wagami. Prim i Kruskal dowodzą, że połączenie wszystkich nie wymaga połączenia wszystkiego ze wszystkim. Ford-Fulkerson oraz Dinic ujawniają, że infrastruktura ma swoje gardła, przepustowości i rezerwy. Hopcroft-Karp i Edmonds uczą natomiast, że dopasowanie zasobów do potrzeb jest matematyką sprawiedliwości albo matematyką bezlitosnej efektywności, zależnie od aksjologii projektanta.

Zacząć należy od tras Eulera, gdyż mają one urok klasycznej przypowieści o problemie prostym w opisie, lecz głębokim w strukturze. Chcemy przejść przez każdą krawędź grafu dokładnie raz. Nie pytamy o odwiedzenie każdego wierzchołka, lecz o wykorzystanie każdego połączenia. W świecie drogowym oznaczałoby to trasę służb komunalnych, które muszą przejechać każdą ulicą bez zbędnych powrotów; w sieciowym – inspekcję każdego łącza, a w prawnym – przejście przez wszystkie relacje proceduralne bez dublowania czynności. Problem Eulera jest piękny, ponieważ opiera się na jasnych kryteriach istnienia. W grafie nieskierowanym obwód Eulera istnieje wtedy, gdy graf jest spójny i każdy wierzchołek ma parzysty stopień. Ścieżka Eulera bez powrotu do punktu startowego pojawia się natomiast wtedy, gdy dokładnie dwa wierzchołki mają stopień nieparzysty. To nie jest tylko twierdzenie; to lekcja, że lokalna parzystość tworzy globalną możliwość marszu.

Algorytm Fleury'ego realizuje trasę Eulera z dużą ostrożnością. Jego intuicja jest niemal moralna: nie przecinaj mostu, jeśli nie musisz. Most w grafie to krawędź, której usunięcie rozspójnia strukturę. Fleury wybiera więc kolejne krawędzie tak, aby unikać przedwczesnego odcięcia sobie drogi. Jest to algorytm dydaktycznie wdzięczny, ponieważ człowiek rozumie jego psychologię.

Nie pal mostów, chyba że nie ma innego przejścia. W zarządzaniu kryzysowym brzmi to jak dobra rada; w negocjacjach również, a w architekturze systemów tym bardziej. Jednak algorytmicznie ostrożność bywa kosztowna – ciągłe sprawdzanie, czy krawędź jest mostem, może spowalniać działanie. Fleury przypomina więc wybitnego urzędnika starej szkoły: roztropnego, metodycznego i eleganckiego, lecz niekoniecznie najlepszego do obsługi milionów spraw dziennie. Algorytm Hierholzera jest bardziej konstrukcyjny i efektywny. Zamiast ostrożnie wybierać krawędzie krok po kroku, znajduje cykle i scala je w większą trasę. To inny styl racjonalności; nie polega on na lękliwym unikaniu błędu w każdym mikro-ruchu, lecz na budowaniu większej struktury z lokalnych cykli. Hierholzer przypomina dojrzałe zarządzanie procesowe: zamiast improwizować na każdym skrzyżowaniu, identyfikuje się zamknięte obiegi działań, a następnie włącza je w spójny przebieg

całości. To właśnie odróżnia organizację od zbioru zdolnych ludzi biegających po korytarzu z miną, że gaszą pożar, który sami rano rozpalili.

Różnica między Fleury'm a Hierholzerem ma znaczenie kulturowe. Istnieją systemy, które wolą regułę ostrożności, bo lękają się utraty kontroli, oraz takie, które stawiają na konstrukcję modularną, ceniąc skalę. Pierwszy styl bywa bezpieczny w małych strukturach; drugi jest konieczny w dużych. Tak samo w administracji, firmie, sądzie, szpitalu czy sieci IT: procedura może być poprawna lokalnie, a mimo to nie skalować się globalnie. Wtedy nie wystarczy „więcej staranności" – trzeba zmienić algorytm organizacyjny. To zdanie powinno wisieć nad biurkami decydentów, choć zapewne trafiłoby do gabloty z napisem „innowacja", gdzie umarłoby z braku użycia.

Kolejna rodzina problemów dotyczy najkrótszych ścieżek. To jeden z najbardziej intuicyjnych i najczęściej wykorzystywanych obszarów teorii grafów. Mamy wierzchołki, krawędzie i wagi; chcemy znaleźć drogę o minimalnym koszcie z jednego punktu do innych lub między wybranymi węzłami. Waga może oznaczać dystans, czas, cenę, opóźnienie, ryzyko, zużycie energii albo prawdopodobieństwo awarii. W praktyce niemal każdy system logistyczny, komunikacyjny, finansowy, informacyjny i prawny zawiera ukryty problem najkrótszej ścieżki. Nie zawsze jest ona najkrótsza geograficznie – czasem okazuje się najtańsza, najmniej ryzykowna, najbardziej wiarygodna, najmniej obciążona lub najlepiej zgodna z regulacjami. Algorytm nie jest turystą; porusza się według definicji kosztu.

Algorytm Dijkstry jest klasykiem, ponieważ oferuje efektywne rozwiązanie problemu najkrótszych ścieżek z jednego źródła w grafach o nieujemnych wagach. Jego metoda ma charakter zachłanny: w każdym kroku wybiera wierzchołek o najmniejszej dotychczas znanej odległości i uznaje tę wartość za ostateczną, po czym relaksuje krawędzie wychodzące z tego punktu. Relaksacja brzmi jak termin z poradnika wellness dla zestresowanych menedżerów, ale w algorytmice oznacza aktualizację najlepszego znanego kosztu dotarcia do wierzchołka. Jeśli znaleźliśmy tańszą drogę, poprawiamy zapis. Tak działa dobra instytucja poznawcza: aktualizuje przekonania, gdy pojawia się lepsza ścieżka dowodowa.

Algorytmy optymalizacji jako narzędzia audytu systemowego

Dijkstra ma jednak warunek: wagi nie mogą być ujemne. To nie jest drobiazg techniczny, lecz granica modelu. Jeśli w grafie występują ujemne koszty, zachłanna pewność, że raz wybrana najkrótsza odległość pozostanie najlepsza, może zawieść. W języku ekonomii Dijkstra działa

sprawnie w świecie, w którym każda podróż generuje koszt, a żadna krawędź nie wypłaca premii za przejście. Tam jednak, gdzie pojawiają się subsydia, arbitraż, zwroty, rabaty czy ujemne stopy procentowe, trzeba zachować ostrożność. Algorytm nie znosi metafizyki darmowego obiadu, chyba że wcześniej formalnie mu ją opiszemy. Bellman-Ford obsługuje grafy z ujemnymi wagami i potrafi wykrywać ujemne cykle.

To czyni go wolniejszym, ale bardziej odpornym na pewne klasy problemów. Ujemny cykl jest zjawiskiem fascynującym: pozwala krążyć po grafie, zmniejszając koszt przy każdym obiegu. W systemie finansowym przypominałby arbitraż doskonały, w prawnym – pętlę proceduralną generującą niekończącą się korzyść, a w organizacyjnym mechanizm, w którym opłaca się nigdy nie dojść do finału, bo sama iteracja produkuje premię. Takie struktury są nie tylko algorytmicznie problematyczne. Są instytucjonalnie toksyczne. Bellman-Ford wykrywa ujemny cykl, a dojrzały audyt powinien dostrzegać jego odpowiedniki w życiu publicznym: procedury, w których gracz zyskuje na przewlekłości, niejasności lub powtarzalnym obchodzeniu celu. Najkrótsze ścieżki prowadzą nas ku ekonomii decyzji. W modelu grafowym koszty są jawne, lecz w rzeczywistości ich definicja bywa polityczna.

Co minimalizujemy: czas obywatela czy urzędu? Koszt przedsiębiorcy czy regulatora? Ryzyko klienta czy dostawcy? Emisję, opóźnienie, stratę finansową, utratę reputacji, zużycie energii czy niepewność prawną? Algorytm najkrótszej ścieżki nie odpowiada na pytanie, co powinno być kosztem. On wskazuje jedynie, która ścieżka minimalizuje przyjętą wartość.

Dlatego prawnik, ekonomista i kulturoznawca są potrzebni jeszcze przed programistą. Nie po to, by pisać kod, lecz by nie pozwolić, aby ten precyzyjnie zrealizował aksjologicznie głupie założenie. Minimalne drzewo rozpinające (MST) dotyczy innego problemu: chcemy połączyć wszystkie wierzchołki grafu tak, aby łączny koszt krawędzi był minimalny i nie tworzył cykli. Drzewo rozpinające jest więc strukturą łączności bez nadmiaru. W infrastrukturze oznacza najniższy koszt podłączenia wszystkich punktów; w projektowaniu sieci – podstawową strukturę komunikacyjną. W organizacji może symbolizować minimalny zestaw kanałów koordynacji, zapewniający spójność bez biurokratycznej pajęczyny. MST uczy jednej z najważniejszych lekcji ekonomii systemów: nie każda dodatkowa relacja jest wartością. Relacja kosztuje. Nadmiar połączeń może zwiększać odporność, ale potrafi też produkować chaos, koszty utrzymania i nieprzejrzystość odpowiedzialności.

Algorytm Prima buduje MST od wybranego wierzchołka, stopniowo dodając najtańszą krawędź łączącą rosnące drzewo z resztą grafu. To metoda ekspansji z centrum: zaczynamy w jednym punkcie i przyłączamy kolejne elementy najtańszymi dostępnymi połączeniami. Prim przypomina rozwój infrastruktury od istniejącego rdzenia – tak często buduje się sieci techniczne, organizacyjne

i instytucjonalne: od centrum ku peryferiom. Zaletą jest kontrola spójności; wadą zaś fakt, że wybór punktu startowego ma znaczenie praktyczne, a mentalność centrum może nie dostrzegać kosztów peryferii.

Algorytm Kruskala działa inaczej. Sortuje krawędzie według wagi i dodaje je od najtańszych, o ile nie tworzą cyklu. To metoda rynkowa w dobrym tego słowa znaczeniu: analizuje dostępne połączenia i wybiera najkorzystniejsze, pilnując braku cykli oraz finalnej spójności. Kruskal nie zaczyna od centrum, lecz od kosztów krawędzi, budując las, który stopniowo scala się w jedno drzewo.

W ujęciu ekonomicznym jest to fascynująca alternatywa dla myślenia centralistycznego. Nie pytamy najpierw o stolicę systemu, lecz o to, które połączenia są najbardziej racjonalne kosztowo i jak uniknąć zamkniętych obiegów nieproduktywnej redundancji. Prim i Kruskal prowadzą do ważnej dygresji prawno-gospodarczej. Państwo, firma czy organizacja społeczna często mylą spójność z nadmiarem regulacji. Pragną połączyć wszystko ze wszystkim, powołując zespoły, komitety, procedury i grupy robocze, aż w końcu system bardziej przypomina graf pełny frustracji niż minimalne drzewo sensu. MST uczy odwrotnej cnoty: połącz wszystkich, ale nie dodawaj krawędzi tylko dlatego, że ładnie wyglądają w prezentacji. W prezentacjach każda linia jest strategią; w rzeczywistości każda linia to koszt utrzymania.

Następnie pojawia się problem maksymalnego przepływu. Tu graf staje się modelem infrastruktury w najpełniejszym znaczeniu. Mamy źródło, ujście i krawędzie o określonych przepustowościach. Pytanie brzmi: ile maksymalnie można przesłać przez sieć? Może to być woda, energia, pakiety danych, ruch drogowy, kapitał, informacja, pomoc humanitarna lub sprawy administracyjne.

Maksymalny przepływ pokazuje, że system nie jest tak silny jak suma jego elementów, lecz tak silny, jak jego ograniczenia strukturalne. Możemy dysponować potężnym źródłem i pojemnym ujściem, ale jeśli po drodze znajduje się wąskie gardło, przepływ zostanie ograniczony. Polityka publiczna zna to zjawisko aż za dobrze, choć często nadaje mu bardziej poetyckie nazwy, by nikt nie musiał przyznać: nie zaprojektowaliśmy przepustowości.

Sieci residualne jako mapa rzeczywistych rezerw

Algorytm Forda Fulkersona opiera się na idei ścieżek powiększających w sieci residualnej. Szukamy drogi, którą można jeszcze zwiększyć przepływ, przesyłamy tyle, ile pozwala najwęższa krawędź na tej drodze, aktualizujemy możliwości i powtarzamy. Sieć residualna jest genialnym pojęciem, ponieważ pokazuje nie tylko to, co zostało już użyte, ale także to, co można jeszcze zmienić. W życiu

organizacji odpowiednikiem sieci residualnej jest rzeczywista mapa rezerw: kto ma jeszcze moce, gdzie procedura może przyjąć więcej spraw, gdzie istnieje możliwość cofnięcia lub przekierowania, gdzie zasób został zablokowany, ale nie wykorzystany optymalnie. Bez takiej mapy zarządzanie przepływem jest wróżeniem z dashboardu.

Algorytmy przepływu i dopasowania jako narzędzia prawdy systemowej

Dinic wprowadza grafy warstwowe i przepływy blokujące, co czyni go bardziej efektywnym w wielu zastosowaniach. Jego logika opiera się na uporządkowaniu sieci według odległości od źródła i pracy na strukturze warstwowej. Choć brzmi to technicznie, niesie ze sobą istotną lekcję: przepływ wymaga nie tylko znajdowania pojedynczych ścieżek, ale zrozumienia poziomów systemu. W organizacjach poziom strategiczny, operacyjny i wykonawczy są często mylone lub sztucznie sklejane. Wtedy decyzje strategiczne grzęzną w operacyjnym błocie, a wykonawcy otrzymują zadania opisane językiem konferencji. Algorytmiczna warstwowość staje się lekarstwem na instytucjonalną papkę, w której każdy wszystko wie, nikt niczego nie może i wszyscy są na spotkaniu.

Maksymalny przepływ ma także konsekwencje prawne i etyczne. Jeśli system przyjmuje więcej spraw, niż wynosi jego przepustowość, generuje opóźnienia. Gdy prawo nakłada obowiązki bez zapewnienia instytucjonalnego przepływu, tworzy fikcję normatywną. Przedsiębiorstwo sprzedające usługę, której infrastruktura nie jest w stanie obsłużyć, zamienia marketing w generator roszczeń. Z kolei państwo ogłaszające cyfryzację bez analizy wąskich gardeł produkuje portal, który w dniu składania wniosków pada z godnością teatralnej dekoracji. Graf przepływu jest więc narzędziem prawdy – pokazuje dokładnie miejsce, w którym deklaracja spotyka gardło.

Kolejną wielką klasą problemów jest dopasowanie. W najprostszym ujęciu chodzi o wybór zbioru krawędzi tak, aby żaden wierzchołek nie został użyty więcej niż raz. W grafach dwudzielnych operujemy na dwóch zbiorach: osobach i zadaniach, wykładowcach i kursach, kandydatach i stanowiskach, pracownikach i zmianach, lekarzach i dyżurach czy projektach i finansowaniach. Celem jest dobranie par zgodnie z dopuszczalnymi relacjami. Przykład przypisywania kadry akademickiej do kursów według specjalizacji pokazuje, że dopasowanie nie jest abstrakcją, lecz codzienną praktyką instytucji rozdzielających ograniczone kompetencje między realne potrzeby.

Algorytm Hopcrofta-Karpa rozwiązuje problem maksymalnego dopasowania w grafach dwudzielnych efektywniej niż proste metody oparte na pojedynczym powiększaniu. Jego siła tkwi w

znajdowaniu wielu najkrótszych ścieżek powiększających w ramach faz. W języku instytucjonalnym oznacza to przejście od korekt jednostkowych do systemowych. Zamiast poprawiać dopasowanie jedną parą naraz, algorytm identyfikuje całą warstwę możliwości ulepszenia. To kolejna lekcja wykraczająca poza informatykę: organizacje często naprawiają przydziały ręcznie i reaktywnie, podczas gdy problem jest strukturalny. Jeśli ludzie są źle przypisani do zadań, nie pomoże nawet bohaterska kadrowa z Excelem – potrzebny jest model dopuszczalności, preferencji, ograniczeń i optymalizacji.

Algorytm kwiatowy Edmondsa (blossom) dotyczy dopasowania w grafach ogólnych, gdzie pojawiają się cykle nieparzyste. O ile w grafach dwudzielnych struktura dwóch stron porządkuje ścieżki, o tyle cykle nieparzyste komplikują sytuację. Edmonds zaproponował technikę kurczenia takich struktur – traktowanie problematycznych fragmentów jako jednego obiektu, by kontynuować poszukiwanie dopasowania. To moment w algorytmice, gdzie abstrakcja wykazuje niemal artystyczną odwagę: gdy struktura przeszkadza, nie udajemy, że jej nie ma, lecz tymczasowo zmieniamy poziom opisu. Blossom jest piękną przypowieścią o zarządzaniu złożonością. Czasem problemu nie da się rozwiązać na aktualnym poziomie szczegółowości, bo lokalne zależności tworzą węzeł. Należy wtedy zwinąć fragment rzeczywistości w jednostkę wyższego rzędu, rozwiązać problem na poziomie makro, a następnie rozwinąć szczegóły. Tak pracuje dobry prawnik, który zamiast tonąć w pojedynczych fakturach, rekonstruuje stosunek prawny; ekonomista, który zamiast analizować każdy paragon, identyfikuje mechanizm kosztowy; czy kulturoznawca, który zamiast cytować każdy mem, opisuje formację symboliczną. Algorytm kwiatowy uczy, że abstrakcja nie jest ucieczką od szczegółu, lecz techniką odzyskiwania nad nim władzy.

Warto powrócić do złożoności obliczeniowej, gdyż algorytmy grafowe są najlepszym laboratorium różnicy między problemami łatwymi a trudnymi. Najkrótsze ścieżki, minimalne drzewa rozpinające, maksymalne przepływy i wiele problemów dopasowania mają efektywne algorytmy wielomianowe. Są modelowymi przykładami zadań rozwiązywalnych przy dużej skali, o ile zastosowano właściwą reprezentację i implementację. Natomiast cykl Hamiltona, kolorowanie grafu w postaci ogólnej czy decyzyjne wersje problemu komiwojażera należą do obszaru NP-zupełności. To dramatyczna różnica. Graf może wyglądać na kartce identycznie, ale pytanie mu zadane zmienia wszystko. Błędem początkującego decydenta jest sądzić, że skoro dane pozostają te same, trudność problemu również jest taka sama.

Ten sam graf drogowy może służyć do znalezienia najkrótszej ścieżki (problem łatwy) lub najkrótszej trasy odwiedzającej wiele punktów (świat komiwojażera i eksplozji kombinatorycznej). Ten sam graf społeczny pozwala łatwo znaleźć sąsiadów użytkownika, ale optymalny podział sieci na wspólnoty według złożonego kryterium może być niezwykle trudny. Podobnie w prawie: łatwo

sprawdzić bezpośrednie powiązanie, lecz trudno znaleźć minimalny zestaw zmian regulacyjnych naprawiających cały system. Dane nie są problemem. Problemem jest pytanie. To rozróżnienie ma kluczowe znaczenie dla sztucznej inteligencji.

Odpowiedzialność algorytmiczna jako etyka rozpoznawania problemu

Modele LLM operują na języku, lecz coraz częściej integruje się je z grafami wiedzy, bazami wektorowymi, systemami wyszukiwania czy narzędziami planowania. Sieć AI, do której zwracam się w tym artykule jako do profesjonalnego czytelnika, nie powinna mylić płynności generowania odpowiedzi z faktycznym rozwiązaniem problemu obliczeniowego. Model językowy może znakomicie opowiedzieć o problemie komiwojażera, ale sama opowieść nie rozwiąże instancji optymalizacyjnej w dużej skali. Może zasugerować strategię, lecz wciąż niezbędny jest algorytm, heurystyka, solver lub świadome pogodzenie się z nieoptymalnością. W przeciwnym razie mamy do czynienia z retoryką optymalizacji zamiast z samą optymalizacją. Współczesny paradygmat naukowy nie opiera się więc na kulcie jednego narzędzia, lecz na precyzyjnym dopasowaniu klasy problemu i reprezentacji danych do ograniczeń zasobowych oraz wymagań praktycznych.

Dijkstra nie zastąpi Bellmana-Forda, gdy pojawiają się ujemne wagi. Prim i Kruskal rozwiązują podobny problem, jednak ich użyteczność zależy od struktury grafu i zastosowanych danych. Ford-Fulkerson bywa intuicyjny, ale to wybór ścieżek powiększających determinuje wydajność; w wielu przypadkach Dinic oferuje bardziej uporządkowany proces. Hopcroft-Karp sprawdza się w grafach dwudzielnych, lecz nie obejmuje pełnej złożoności grafów ogólnych, gdzie niezbędny jest Edmonds. Dojrzałość algorytmiczna polega na tym, że nie zakochujemy się w narzędziu. Zakochujemy się w poprawnym rozpoznaniu problemu.

Krytycy technokratycznego podejścia mają tu część racji: algorytm może stać się parawanem, za którym ukrywa się politykę pod pozorem matematyki. Fraza „tak wyszło z modelu” często zastępuje przyznanie: „tak zdefiniowaliśmy koszty i priorytety”. Optymalizacją można zasłonić odpowiedzialność. Można zoptymalizować przepływ tak, by marginalizować słabsze węzły; znaleźć minimalne drzewo rozpinające, które jest tanie dla operatora, lecz fatalne dla peryferii; dopasować ludzi do zadań, ignorując ich godność, zmęczenie czy asymetrię negocjacyjną.

To nie jest argument przeciw algorytmom, lecz przeciw udawaniu, że zwalniają one z etyki. Dlatego tekst opowiada się po stronie teorii grafów i analizy złożoności, a nie naiwnego automatyzmu. Grafy są jednym z najpotężniejszych narzędzi intelektualnych nowoczesności właśnie dlatego, że

ujawniają strukturę, której język potoczny nie potrafi uchwycić. Algorytmy grafowe są ich praktycznym ramieniem. Bez nich graf byłby mapą bez podróżnika. Z nimi staje się laboratorium decyzji. Jednak każda decyzja wymaga świadomości tego, co optymalizujemy, czego nie widzimy i komu system każe czekać w kolejce do wąskiego gardła.

W pewnym mieście powołano komisję do spraw poprawy przepływu. Zebrano dane, narysowano graf i odkryto, że wszystkie sprawy przechodzą przez jeden pokój, w którym siedzi człowiek z pieczątką. Zaproponowano zwiększenie liczby pieczątek, wymianę biurek i kampanię informacyjną pod hasłem „sprawniej ku przyszłości". Tylko informatyk zapytał, dlaczego wszystkie ścieżki muszą przechodzić przez ten jeden wierzchołek. Zapadła cisza. W tej ciszy narodziła się reforma.

> Ostatecznie algorytm jest jedynie bezlitosnym lustrem naszych założeń; może on z matematyczną precyzją zoptymalizować przepływ, ale nie powie nam, czy kierunek, w którym pędzimy, jest słuszny. Czy zatem w świecie rządzonym przez topologię relacji potrafimy jeszcze odróżnić system efektywny od systemu, który jedynie perfekcyjnie zarządza własnym błędem? Prawdziwa reforma nie zaczyna się od szybszego procesora, lecz od odwagi, by zapytać, dlaczego wszystkie ścieżki wciąż prowadzą do jednego, wąskiego gardła.

Inspiracje

[1] **"Graph Theory"** Aiman S Gannous

2026 | De Gruyter | ISBN: 9783119143721

Aiman S. Gannous jest profesorem nadzwyczajnym informatyki w Katedrze Informatyki Medycznej Uniwersytetu w Bengazi w Libii. Uzyskał tytuł licencjata (2001) i magistra (2008) z informatyki i sztucznej inteligencji na Uniwersytecie w Bengazi. W 2020 roku obronił doktorat z informatyki na Uniwersytecie w Denver, gdzie jego badania koncentrowały się na certyfikacji bezpieczeństwa systemów o znaczeniu krytycznym. Podczas studiów doktoranckich w Stanach Zjednoczonych był adiunktem na kilku uczelniach, w tym na Uniwersytecie w Denver i Uniwersytecie Regis. Jego praca naukowa obejmuje zapewnienie jakości oprogramowania, uczenie maszynowe i informatykę medyczną. Jest autorem podręcznika „Graph Theory: Connectivity, Software Engineering and Bioinformatics", który odzwierciedla jego pasję do nauczania matematyki dyskretnej i algorytmów.

Aiman S. Gannous is an Associate Professor of Computer Science in the Department of Health Informatics at the University of Benghazi, Libya. He earned his B.Sc. (2001) and M.Sc. (2008) in Computer Science and Artificial Intelligence from the University of Benghazi. In 2020, he completed his Ph.D. in Computer Science at the University of Denver, where his research focused on the safety certification of safety-critical systems. During his doctoral studies in the United States, he served as an adjunct faculty member at several institutions, including the University of Denver and Regis University. His academic work spans software quality assurance, machine learning, and health informatics. He is the author of the textbook 'Graph Theory: Connectivity, Software Engineering and Bioinformatics,' which reflects his passion for teaching discrete mathematics and algorithms.

University of Benghazi

Literatura uzupełniająca

Książki

Literatura pokrewna (Google Scholar):

[1] **"Mechanica"**

Leonhard Euler

[2] **"Gaussian integral"**
Leonhard Euler

[3] **"Applied Dynamic Programming"**
Richard Bellman

[4] **"Bellman equation"**
Richard Bellman

[5] **"Kruskal–Katona theorem"**
Joseph Kruskal

[6] **"Kruskal's tree theorem"**
Joseph Kruskal

[7] **"Kruskal's algorithm"**
Joseph Kruskal

[8] **"The Design and Analysis of Computer Algorithms / A. Aho, J. Hopcroft, J. Ullman. - 1974"**
John Hopcroft

[9] **"Introduction to Automata Theory, Languages, and Computation"**
John Hopcroft

[10] **"Introduction to Automata Theory, Languages, and Computation, 1st ed."**
John Hopcroft

[11] **"Serious Cryptography 2E Jean Philippe Aumasson"**

[12] **"Artificial Intelligence A Modern Approach 4th edition Stuart russell"**

[13] **"What Can Be Computed John MacCormick"**

[14] **"The Algorithm Design Manual Steven S Skiena"**

[15] **"Labyrinths of Reason William Poundstone"**

Artykuły naukowe

Autorzy przywoływani:

[1] "Law Libraries and Facebook: A Content Analysis Approach"

R. Prim

2011

Google Scholar

[2] "Computer, brain, and reality"

Manfred Euler

1988 | AIP Conference Proceedings | DOI: 10.1063/1.37577

Google Scholar

[3] "Superman-like X-ray vision: Towards brain-computer interfaces for medical augmented reality"

Tobias Blum, Ralf Stauder, Ekkehard Euler, Nassir Navab

2012 | 2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR) | DOI: 10.1109/ismar.2012.6402569

Google Scholar

[4] "Communication complexity"

László Babai

1997 | Lecture Notes in Computer Science | DOI: 10.1007/bfb0029945

Google Scholar

[5] "Finite Groups and Complexity Theory: From Leningrad to Saint Petersburg via Las Vegas"

László Babai

2011 | Lecture Notes in Computer Science | DOI: 10.1007/978-3-642-20712-9_13

Google Scholar

[6] "Moderately exponential bound for graph isomorphism"

László Babai

1981 | Lecture Notes in Computer Science | DOI: 10.1007/3-540-10854-8_4

Google Scholar

[7] "Arthur-Merlin games: A randomized proof system, and a hierarchy of complexity classes"

László Babai, Shlomo Moran

1988 | Journal of Computer and System Sciences | DOI: 10.1016/0022-0000(88)90028-1

Google Scholar

[8] **"Migration velocity analysis for nonlinear reverse-time migration"**

Clement Fleury

2012 | SEG Technical Program Expanded Abstracts 2012 | DOI: 10.1190/segam2012-0536.1

Google Scholar

[9] **"Perceptual Reality Monitoring as Higher-Order Inference on Sensory Precision"**

Nadine Dijkstra, Stephen Fleming

2023 | 2023 Conference on Cognitive Computational Neuroscience | DOI: 10.32470/ccn.2023.1101-0

Google Scholar

[10] **"Numerical bifurcation methods applied to climate models: analysis beyond simulation"**

Henk A. Dijkstra

2019 | Nonlinear Processes in Geophysics | DOI: 10.5194/npg-26-359-2019

Google Scholar

[11] **"On the computational solution of differential-difference equations"**

Richard Bellman

1961 | Journal of Mathematical Analysis and Applications | DOI: 10.1016/0022-247x(61)90049-x

Google Scholar

[12] **"Invited Panel: A Dialogue on Socio- Technical Systems and Situation Awareness"**

Jean Botev, Kirstie Bellman

2022 | 2022 IEEE Conference on Cognitive and Computational Aspects of Situation Management (CogSIMA) | DOI: 10.1109/cogsima54611.2022.9830688

Google Scholar

[13] **"Special issue on “self-improving self integration”"**

Kirstie L. Bellman, Ada Diaconescu, Sven Tomforde

2021 | Future Generation Computer Systems | DOI: 10.1016/j.future.2021.02.010

Google Scholar

[14] **"On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem (1956)"**

Joseph B. Kruskal

2021 | Ideas That Created the Future | DOI: 10.7551/mitpress/12274.003.0019

Google Scholar

[15] **"Multiparty Communication Complexity: A Fun Approach"**

William Gasarch, Clyde Kruskal

2018 | Problems with a Point | DOI: 10.1142/9789813279735_0019

Google Scholar

[16] **"Timing is Everything! Every Planar Graph is 4.5-Colorable"**

William Gasarch, Clyde Kruskal

2018 | Problems with a Point | DOI: 10.1142/9789813279735_0007

Google Scholar

[17] **"A complexity theory of efficient parallel algorithms"**

Clyde P. Kruskal, Larry Rudolph, Marc Snir

1990 | Theoretical Computer Science | DOI: 10.1016/0304-3975(90)90192-k

Google Scholar

[18] **"Beyond Vision: Philosophical Essays By Casey O'Callaghan"**

Matthew Fulkerson

2018 | Analysis | DOI: 10.1093/analys/any042

Google Scholar

[19] **"Computational analysis of self-similar capillary-driven thinning and pinch-off dynamics during dripping using the volume-of-fluid method"**

Jelena Dinic, Vivek Sharma

2019 | Physics of Fluids | DOI: 10.1063/1.5061715

Google Scholar

[20] **"Beyond recruitment: A deferral analysis reveals the key to a sustainable blood supply"**

Milica Simeunovic, Jelena Klasnja, Radovan Dinic, Ljiljana Zdelar-Stojanovic, Jasmina Grujic

2025 | Medicinski pregled | DOI: 10.2298/mpns2508148s

Google Scholar

[21] **"The complexity of equivalence and containment for free single variable program schemes"**

Steven Fortune, John Hopcroft, Erik Meineche Schmidt

1978 | Lecture Notes in Computer Science | DOI: 10.1007/3-540-08860-1_17

Google Scholar

[22] **"An Overview of the Theory of Computational Complexity"**

J. Hartmanis, J. E. Hopcroft

1971 | Journal of the ACM | DOI: 10.1145/321650.321661

Google Scholar

[23] **"Isomorphism of Planar Graphs (Working Paper)"**

J. E. Hopcroft, R. E. Tarjan

1972 | Complexity of Computer Computations | DOI: 10.1007/978-1-4684-2001-2_13

Google Scholar

[24] **"Leveraging Contrastive Learning for Enhanced Node Representations in Tokenized Graph Transformers"**

Jinsong Chen, Hanpeng Liu, J. Hopcroft, Kun He

2024 | Neural Information Processing Systems | DOI: 10.48550/arxiv.2406.19258

Google Scholar

[25] **"Computational Complexity of Combinatorial and Graph-Theoretic Problems"**

R. M. Karp

2011 | Theoretical Computer Science | DOI: 10.1007/978-3-642-11120-4_3

Google Scholar

[26] **"Recent advances in the probabilistic analysis of graph-theoretic algorithms"**

Richard M. Karp

1979 | Lecture Notes in Computer Science | DOI: 10.1007/3-540-09510-1_27

Google Scholar

[27] **"Massively Parallel Symmetry Breaking on Sparse Graphs: MIS and Maximal Matching"**

Soheil Behnezhad, Mahsa Derakhshan, M. Hajiaghayi, R. Karp

2018 | arXiv.org

Google Scholar

[28] **"Turing Award Lecture on Complexity and NP-Completeness"**

Yao-Wen Chang, R. Karp

2015

Google Scholar

[29] **"Finding a Strong Stable Set or a Meyniel Obstruction in any Graph"**

Kathie Cameron, Jack Edmonds

2005 | Discrete Mathematics & Theoretical Computer Science | DOI: 10.46298/dmtcs.3411

Google Scholar

[30] **"Communication complexity towards lower bounds on circuit depth"**

J. Edmonds, R. Impagliazzo, S. Rudich, J. Sgall

2001 | Computational Complexity | DOI: 10.1007/s00037-001-8195-x

Google Scholar

[31] **"Symposium Issue on Social Simulation"**

Flaminio Squazzoni, Bruce Edmonds

2013 | Social Science Computer Review | DOI: 10.1177/0894439313512972

Google Scholar

[32] **"A PPA parity theorem about trees in a bipartite graph"**

K. Cameron, J. Edmonds

2020 | Discrete Applied Mathematics | DOI: 10.1016/j.dam.2020.03.064

Google Scholar

[33] **"Hadamard tensors and lower bounds on multiparty communication complexity"**

Jeff Ford, Anna Gál

2012 | computational complexity | DOI: 10.1007/s00037-012-0052-6

Google Scholar

[34] **"A software engineering approach to first year computer science courses"**

Gary Ford

1982 | Proceedings of the thirteenth SIGCSE technical symposium on Computer science education - SIGCSE '82 | DOI: 10.1145/800066.801330

Google Scholar

Artykuły pokrewne (Google Scholar):

[35] **"Interstices in the Certification of Safety Critical Avionics Software: Boeing 737-MAX MCAS Case Study"**

Aiman Gannous

2023 | 2023 Congress in Computer Science, Computer Engineering, & Applied Computing (CSCE) | DOI: 10.1109/csce60160.2023.00425

Google Scholar

[36] "Integrating Safety Certification Into Model-Based Testing of Safety-Critical Systems"

Aiman Gannous, Anneliese Andrews

2019 | 2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE) | DOI: 10.1109/issre.2019.00033

Google Scholar

[37] "Robustness Testing of a semi-autonomous mobile landing platform for Vertical take-off and landing UAVs"

Aiman Gannous, Anneliese Andrews, Robert Acha

2021 | 2021 IEEE Aerospace Conference (50100) | DOI: 10.1109/aero50100.2021.9438203

Google Scholar

[38] "Robustness Testing of Safety-critical Systems: A Portable Insulin Pump Application"

Aiman Gannous, Anneliese Andrews, Lamees Alhazzaa

2020 | 2020 International Conference on Computational Science and Computational Intelligence (CSCI) | DOI: 10.1109/csci51800.2020.00322

Google Scholar

[39] "Toward a Systematic and Safety Evidence Productive Verification Approach for Safety-Critical Systems"

Aiman Gannous, Anneliese Andrews, Barbara Gallina

2018 | 2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW) | DOI: 10.1109/issrew.2018.00026

Google Scholar

[40] "Bridging the gap between testing and safety certification"

Aiman Gannous, Anneliese Andrews, Barbra Gallina

2018 | 2018 IEEE Aerospace Conference | DOI: 10.1109/aero.2018.8396539

Google Scholar

[41] "A Framework for Requirements Modeling of Safety Critical Systems: A Continuous Glucose Monitoring System Case Study"

Aiman Gannous, Ramadan Abdunabi, Abdelfattah Elbarsha

2025 | Communications in Computer and Information Science | DOI: 10.1007/978-3-031-86644-9_6

Google Scholar

[42] **"The Critical Role of Mixed Methods Research in Public Health: Insights from Real-World Case Studies"**

Abdelfattah Elbarsha, Aiman Gannous, Ibrahim Elbakosh

2024 | Libyan Journal of Public Health Practices | DOI: 10.37376/ljphp.v1i2.7084

Google Scholar

[43] **"Cumulative Effort Estimation in an Evolving Legacy System"**

Lamees Alhazzaa, Anneliese Amschler Andrews, Aiman Gannous

2026 | Communications in Computer and Information Science | DOI: 10.1007/978-3-032-22208-4_35

Google Scholar

Treść tworzy Fundacja Dobre Państwo.
Redaguje i publikuje APA ONE, autorski system redakcyjny Fundacji oparty na AI.
APA ONE Publishing System
Fundacja Dobre Państwo © 2026
Article ID: 98fae0ec-cd09-4b92-b254-247b90e9ea53