Lamiyə Rafiq MƏMMƏDZADƏ
Qərbi Kaspi Universiteti, Yüksək texnologiyalar və innovativ həllər məktəbinin magistrantı
E-mail: lamiye.qku@gmail.com

QRAF ALQORİTMLƏRİNİN TƏDQİQİ VƏ İNKİŞAFI: TƏTBİQİ PROQRAM MÜHƏNDİSLİYİNDƏ PERSPEKTİVLƏR

Xülasə

Bu tədqiqat işində qraf nəzəriyyəsinin abstrakt riyazi modeldən müasir proqram təminatı mühəndisliyinin fundamental alətinə çevrilməsi prosesi təhlil edilir. Müasir sistemlərin arxitekturasında yaddaşın idarə edilməsi, CPU keş optimallaşdırılması və şəbəkə gecikmələri kimi aparat məhdudiyyətləri qraf emalının səmərəliliyinə birbaşa təsir göstərir. Məqalədə RAM daxilində məlumatların təşkili üsulları, xüsusilə statik qraf strukturları üçün yüksək məhsuldarlıqlı "Compressed Sparse Row" (CSR) metodu araştırılır. Tədqiqat çərçivəsində Dijkstra və A* kimi klassik marşrutlaşdırma alqoritmlərinin müasir modifikasiyaları və onların real mühitdəki tətbiqləri nəzərdən keçirilir. Əlaqəli (relational) verilənlər bazalarının mürəkkəb iyerarxik məlumatlar qarşısındakı limitləri fonunda Qraf Verilənlər Bazası İdarəetmə Sistemlərinin (GDBMS) zəruriliyi əsaslandırılır. Facebook-un TAO arxitekturası nümunəsində milyardlarla obyektin idarə olunması mexanizmləri izah edilir. Həmçinin, "Random Walk" alqoritmlərinin Sybil hücumlarının (saxta hesablar) qarşısının alınmasındakı rolu və qraf neyron şəbəkələrinin (GNN) gələcək perspektivləri, eləcə də FPGA/GPU əsaslı aparat sürətləndiricilərinin əhəmiyyəti vurgulanır.

Açar sözləri: qraf nəzəriyyəsi, hesablama alqoritmləri, məlumat strukturları, qraf yönümlü verilənlər bazaları (GDBMS), sistem etibarlılığı.

UOT: 004.65:004.421
JEL: C63, C88, L86
DOI: https://doi.org/10.54414/CXAP4992

Giriş

Proqram təminatının inkişafının müasir paradiqmasında qrafik nəzəriyyəsi diskret riyaziyyat kursları daxilində öyrənilən akademik intizam olmaqdan çıxaraq yüksək yüklü sistemlərin layihələndirilməsi üçün kritik alətə çevrildi [11, s.1]. Düyünlərin (təpələrin) və əlaqələrin (kənarların) toplusunu təmsil edən qrafiklər sosial şəbəkələrdən və yol xəritələrindən tutmuş molekulyar strukturlara qədər mürəkkəb real dünya münasibətlərinin modelləşdirilməsi üçün təbii abstraksiyadır. Bununla belə, tətbiqi proqram mühəndisliyində nəzəri modellərdən praktik tətbiqə keçid ciddi problemlərlə müşayiət olunur. Nəzəri alqoritmik mürəkkəblik çox vaxt Big O notasiyası baxımından qiymətləndirilsə də, mühəndislik praktikasında yaddaşın idarə edilməsi, CPU keşində verilənlərin yerləşdirilməsi, paylanmış sistemlərdə şəbəkə gecikməsi və verilənlərin ardıcıllığı məsələləri ön plana çıxır [5, s.45]. Qrafik alqoritmləri üzrə dərin tədqiqatların aktuallığı məlumat həcmlərinin eksponensial artımı ilə müəyyən edilir. Facebook sosial qrafiki və ya Google indeksi kimi müasir qrafiklər milyardlarla təpə və trilyonlarla kənarları ehtiva edir [7, s.494]. Belə strukturların emalı üçün təkcə səmərəli alqoritmlər deyil, həm də məlumatların saxlanması və emalı üçün prinsipcə yeni yanaşmalar tələb olunur. Ənənəvi Relational Database Management Systems (RDBMS) dərin əlaqəli verilənlərlə işləyərkən səmərəsizliyini nümayiş etdirir ki, bu da xüsusi qrafik verilənlər bazalarının və yaddaşdaxili qrafik işlənməsinin yaranmasına səbəb olub [8, s.3].

Bu araşdırma qrafik alqoritminin inkişafının həyat dövrünün hərtərəfli təhlilini təqdim edir: aparat arxitekturası üçün optimal-

laşdırılmış aşağı səviyyəli məlumat strukturlarının seçilməsindən mürəkkəb naviqasiya sistemlərinin və paylanmış yaddaşın tətbiqinə qədər. Nəzəri konsepsiyaların müasir sənayenin tələb etdiyi performans, genişlənmə və etibarlılığın ciddi tələblərinə necə uyğunlaşdığına xüsusi diqqət yetirilir. Biz aparıcı texnoloji şirkətlərin (Facebook-dan istifadə etməklə) yol tapma alqoritmlərinin [2, s.269; 3, s.100], memarlıq həllərinin təkamülünü araşdıracağıq. İstənilən qrafik alqoritminin səmərəliliyi RAM-da qrafiki təmsil etmək üçün istifadə olunan məlumat strukturu ilə ayrılmaz şəkildə bağlıdır. Tətbiqi proqram mühəndisliyində bitişiklik matrisi, bitişiklik siyahısı və ya sıxılmış formatlar (məsələn, CSR) arasında seçim nadir hallarda yalnız asimptotik mürəkkəbliklə diktə edilir [4, s.2]. Sıxılmış Seyrək Sıra (CSR) tərəfindən təmin edilən məkan lokallığı, genişmiqyaslı qrafik keçidlərində ümumi bir maneə olan keş səhvlərinin tezliyini minimuma endirir [4, s.5]. Bunun əvəzinə, bu qərar yaddaş istehlakı, məlumatların yenilənməsi sürəti və ən əsası müasir arxitekturalarda CPU keşinin lokalizasiyası arasında kompromisdir [5, s. 60].

Qonşuluq matrisi: nəzəriyyə reallığa qarşı

Qonşuluq matrisi V sayda V ölçülü 2D massivi kimi G=(V,E) qrafikini təmsil edir, burada (i,j) xanası i və j təpələri arasında kənarın mövcudluğu (və potensial çəkisi) haqqında məlumat ehtiva edir [1, s.590]. Böyük miqyaslı sistemlərdə qonşuluq matrisinin daxili səmərəsizliyi onun O(V^2) fəza mürəkkəbliyindən qaynaqlanır ki, bu da sosial qrafiklər və ya veb indeksləri kimi seyrək real dünya şəbəkələrini modelləşdirərkən qadağanedici hala gəlir [1, s.591]. Mühəndislik praktikasında milyard düyünlü bir qrafiki matris kimi saxlamaq üçün petabayt RAM tələb olunur ki, bunun da əksəriyyəti sıfırlarla doldurulur və bu da yaddaşın həddindən artıq istifadəsinə səbəb olur [8, s.4]. Bundan əlavə, sıx bir matrisdə sətir skanını apararkən, CPU tez-tez lazımsız məlumatları keş xətlərinə gətirir və mövcud olmayan kənarlarda vaxtını boşa xərcləyir [5, s.22].

Bu aparat səviyyəli maneələri aradan qaldırmaq üçün müasir sistemlər Sıxılmış Seyrək Sıra (CSR) təsvirinə keçir. CSR, bitişiklik siyahılarında olan "göstərici təqibi" problemini bütün çıxan kənarları tək, bitişik bir massivə yığmaqla həll edir ki, bu da CPU-nun fəzada effektivliyini maksimum dərəcədə artırır [4, s.7]. Fərdi obyekt başlıqlarının və göstəricilərinin üst hissəsini aradan qaldırmaqla, KSM, təpənin bütün qonşuluğunun tək bir yaddaş dövrəsi ilə L1 keşinə axın etməsinə imkan verir [5, s.64].

Yüksək performanslı hesablama (High performance computing (HPC)) mühitlərində bu sıx məlumat qablaşdırması, GPU yaddaşı və emal bölmələri arasında məlumat hərəkətini minimuma endirən Rabitədən Qaçınma (Communication-Avoiding) alqoritmlərindən istifadə etməyə imkan verir [6, s.12]. Cormen və digərləri tərəfindən vurğulandığı kimi, bitişiklik matrisi bir kənarın (u, v) mövcud olub-olmadığını müəyyən etmək üçün nəzəri cəhətdən üstün olsa da, KSM və bitişiklik siyahılarına praktik keçid, əksər real dünya qrafiklərinin orta dərəcəsinin V-dən daha kiçik olması ilə əlaqədardır [1, s.592]. Nəticə etibarilə, müasir proqram təminatı mühəndisliyinin "Reallığı", kütləvi məkan itkisi bahasına nəzəri O(1) girişi təklif edənlərə nisbətən yaddaş iyerarxiyasına və keş lokallığına təmin edən məlumat strukturlarına üstünlük verir.Teorik olaraq, bu struktur O(1) kənar axtarış vaxtını təmin edərək, sıx qrafiklər üçün idealdır. Klassik obyekt yönümlü tətbiqetmədə hər bir qrafik node yığında ayrılmış ayrıca obyektdir. Qrafik keçidi zamanı prosessor ixtiyari yaddaş ünvanlarına daim daxil olmağa məcbur olur. RAM CPU keşindən (L1, L2, L3) daha yavaş olduğundan, hər bir keş buraxılışı yüzlərlə dövr üçün hesablamanı dayandırır [5, s.30].

Əsas hissə

Yüksək yüklü sistemlərdə, xüsusən oxunması ağır olan sistemlərdə performans problemlərini həll etmək üçün Sıxılmış Seyrək Sıra (CSR) formatından istifadə olunur. CSR bitişik yaddaşda saxlanılan üç birölçülü massivdən istifadə edən qrafiki təmsil edir [4, s.8]:

1. Dəyərlər Massivi: Kənar çəkiləri saxlayır (əgər qrafik çəkilibsə).

2. Sütun İndeksləri Massivi: Hər kənar üçün hədəf təpə identifikatorlarını saxlayır.

3. Sıra Ofsetləri Massivi: Sütun İndeksləri massivində hər bir təpə üçün qonşu siyahısının başlanğıc indekslərini saxlayır.

KSM-nin əsas mühəndislik üstünlüyü məlumatların yerləşməsidir. Təpənin bütün çıxan kənarları ardıcıl olaraq saxlanıldığından, qonşular üzərində təkrarlanan zaman prosessor eyni vaxtda bir qrup kənarları keş xəttinə yükləyir [5, s.15]. Bu, RAM-a girişi minimuma endirir və aparatın əvvəlcədən təyin edilməsindən səmərəli istifadə etməyə imkan verir. Testlər göstərir ki, CSR-dən istifadə statik qrafiklərdə BFS və DFS kimi keçid alqoritmləri üçün bitişiklik siyahıları ilə müqayisədə 3 dəfəyə qədər performans artımı təmin edə bilər [6, s.5]. Buna göstəricilərin və obyekt metaməlumatlarının saxlanması ilə bağlı əlavə xərclərin aradan qaldırılması və daha sıx məlumatların qablaşdırılması ilə nail olunur. Aparat baxımından, KSM-in təbiəti CPU-nun keş xətti ölçüsündən (adətən 64 bayt) istifadə edir. Prosessor bir qonşunun ID-sini yüklədikdə, təsadüfən növbəti bir neçə qonşunu L1 keşinə gətirir və sonrakı iterasiyaların gecikməsini effektiv şəkildə sıfıra yaxınlaşdırır [5, s.32]. Bu, obyekt yönümlü bitişik siyahılarından kəskin bir ziddiyyətdir, burada hər bir qonşu fərqli bir yaddaş səhifəsində yerləşə bilər və bu da araşdırılan hər bir kənar üçün "Stop the world" keşinin səhvini vurğulayır [5, s.45]. Bundan əlavə, KSM, registrlər daxilində məlumatların təkrar istifadəsini maksimum dərəcədə artırmaq üçün hesablama qaydasını yenidən quran "Ünsiyyətdən yayınmaq (Communication Avoiding)" nüvələrinin tətbiqinə imkan verir [6, s.18]. KSM, kənarların tez-tez əlavə edildiyi və ya silindiyi dinamik qrafiklər üçün daha az səmərəli olsa da - bitişik bir massivdə elementlərin dəyişdirilməsinin O(E) dəyəri səbəbindən - Google indeksi və ya əvvəlcədən işlənmiş yol şəbəkələri kimi statik və ya oxuma ağırlığı olan məlumat dəstləri üçün qızıl standart olaraq qalır [4, s.12]. Məlumat strukturunu yaddaş iyerarxiyasının fiziki reallıqları ilə uyğunlaşdırmaqla, tərtibatçılar ənənəvi göstərici əsaslı təmsilçiliklərlə riyazi olaraq qeyri-mümkün olan performans səviyyələrinə nail ola bilərlər [5, s.90]. Müasir C++ və Rust inkişafında Data-Oriented Design paradiqma populyarlıq qazanır, kod strukturundan daha çox məlumat tərtibinə üstünlük verir [5, s.72]. Qrafiklərə tətbiq edilən bu, hər bir qovşağın öz koordinatlarını, çəkisini və vəziyyətini saxladığı "Strukturlar Massivi" (AoS) yanaşmasını "Masivlərin Strukturu"nun xeyrinə rədd etmək deməkdir (SoA).

Yaddaş girişindən başqa, budaq proqnozu kritik amildir. Qrafik alqoritmlər şərti sıçrayışlarla zəngindir (məsələn, yoxlanılması (Ivisited[node]). Bir budağın səhv proqnozlaşdırılması prosessoru təmizləyir. Cachegrind və ya Valgrind kimi alətlərlə profil yaratmaq bu çətinlikləri aşkar edir. Mühəndislər tez-tez budaqsız proqramlaşdırma üsullarına müraciət edirlər və ya giriş nümunələrini aparat üçün daha proqnozlaşdırıla bilən etmək üçün qonşu siyahıları çeşidləyirlər. Məsələn, qonşuluq siyahılarının çeşidlənməsi müəyyən edilmiş kəsişmə əməliyyatlarını sürətləndirə bilər.

Naviqasiya və pathfinding: dijkstradan ierarxik sistemlərə

Yol Axtarışı tətbiqi proqramlaşdırmada ən tələbkar vəzifələrdən biridir. Dijkstranın orijinal 1959-cu il formulasiyası ən qısa yollar üçün qızıl standart olaraq qalsa da, kütləvi qrafiklərdə hesablama xərcləri müasir evristik təkmilləşdirmələr tələb edir [2, s.270].

Pathfinding tətbiqi proqramlaşdırmada ən çox tələb olunan vəzifələrdən biridir.Əsas GPS naviqatorları, logistika sistemləri, kompüter şəbəkəsinin marşrutlaşdırılması və oyun AI [1, s.620].

A* Alqoritmi: evristik optimallaşdırma

Əlamətdar əsaslı evristikalar, A axtarışı daha aşağı sərhədlər təmin etmək üçün üçbucaq bərabərsizliyindən istifadə edir və böyük yol şəbəkələrində tədqiqat sahəsini daha da azaldır [3, s.105]. Statik çəkilərdən başqa, müasir marşrutlaşdırma mühərrikləri real vaxt trafik sıxlaşmalarını nəzərə almaq üçün zamandan asılı kənar xərcləri daxil edir və dinamik qrafik yeniləmələrini tələb edir [1, s.650]. Hub Etiketlərinin (HL) inteqrasiyası, hər bir qovşaq üçün etiketləri əvvəlcədən hesablamaqla qitə miqyaslı şəbəkələrdə mikrosaniyədən aşağı məsafə sorğularına imkan verir [11, s.155].

A* (A-Star) alqoritmi hədəfə qədər qalan məsafəni təxmin edən h(n) evristik funksiyasını əlavə etməklə Dijkstranın fikirlərini təkmilləşdirir [2, s.270]. Node prioriteti f(n) = g(n) + h(n) kimi müəyyən edilir [3, s.102].

Performans: A* axtarış sahəsini əhəmiyyətli dərəcədə azaldaraq, axtarışı məqsədə doğru istiqamətləndirir. Bununla belə, həndəsi cəhətdən ən qısa yolun həmişə ən sürətli olmadığı mürəkkəb yol şəbəkələrində (məsələn, dolama magistral və düz torpaq yol) qeyri-kafi ola bilər. Ölkə və qitə miqyaslı naviqasiya tapşırıqları üçün hətta A* kifayət qədər sürətli deyil [1, s.658]. Müasir naviqasiya mühərrikləri (OSRM, GraphHopper, Google Maps) sıxılma iyerarxiyaları (CH) kimi qrafikin əvvəlcədən işlənməsi üsullarından istifadə edir.

Sıxılma İerarxiyaları (CH): İş prinsipi

CH yol şəbəkəsinin təbii iyerarxiyasından istifadə edir [11, s.150]. Yerli yollar magistral yollara aparan arteriyalara aparır. Alqoritm iki mərhələdən ibarətdir:

1. Əvvəlcədən emal mərhələsi: Qrafik qovşaqlar “əhəmiyyət”ə görə sıralanır. Daha sonra, ən az əhəmiyyətli qovşaqlar iterativ olaraq "büdcələnir". Bir qovşağın çıxarılması onun qonşuları arasındakı ən qısa yolu pozarsa, qrafikə "qısa yol" kənarı əlavə olunur. Bu, sürətli keçidlərlə zənginləşdirilmiş çoxsəviyyəli qrafik yaradır [1, s.670].

2. Sorğu mərhələsi: İki istiqamətli Dijkstra axtarışı həyata keçirilir. İrəli axtarış yalnız daha vacib qovşaqlara aparan kənarları nəzərə alır. Geri izləmə metodu yalnız artan əhəmiyyəti izləyir. Axtarışlar iyerarxiyanın "yuxarısında" görüşür [2, s.271].

Qrafik Model və Əlaqəli model

Qrafiklərlə işləyərkən RDBMS-nin məhdudiyyətləri empedans uyğunsuzluğu kimi tanınır. "Yerli Qrafik Yaddaşı"na keçid, cədvəl skanları əvəzinə çoxlu naviqasiya sorğularını optimallaşdırmaq üçün məlumatların diskdə fiziki olaraq təşkil edilməsini təmin edir [8, s.42]. SQL-də "çoxdan çoxa" münasibətlərinin modelləşdirilməsi cədvəllərin birləşməsini tələb edir. "Dostların dostlarının dostlarını tap" (dərinlik 3) kimi sorğunun yerinə yetirilməsi üçün hesablama dəyəri verilənlərin həcmi ilə eksponent olaraq artan üç qoşulma əməliyyatı tələb olunur. Qrafik verilənlər bazaları indekssiz bitişiklik həyata keçirir, burada hər bir node öz əlaqələri üçün fiziki göstəriciləri ehtiva edir. Bu, ümumi verilənlər bazası ölçüsündən asılı olmayaraq, kənardan keçmə xərclərini sabit edir [8, s.40].

RDF və Əmlak Qrafiklər

Məlumatların qarşılıqlı əlaqəsi üçün Resurs Təsviri Çərçivəsi (RDF) biliyi təmsil etmək üçün standart bir yol təqdim edir, baxmayaraq ki, onun üçlü saxlama arxitekturası real vaxt əməliyyat tətbiqlərində əlavə xərclər tətbiq edə bilər [9, s.15]. Bunun əksinə olaraq, Əmlak Qrafikləri zəngin metaməlumatların birbaşa kənarlarda saxlanmasına imkan verir ki, bu da mürəkkəb tövsiyə mühərrikləri qurmaq üçün daha intuitivdir [8, s.22].

Bazarda iki məlumat modeli üstünlük təşkil edir:

1. RDF (Resurs Təsviri Çərçivəsi): "Mövzu-Predikat-Obyekt" üçlüyü əsasında. Semantik Veb üçün W3C standartı. SPARQL sorğu dilindən istifadə edir. Məlumat inteqrasiyası və nəticə çıxarma işlərində güclüdür, lakin əməliyyat tətbiqləri üçün geniş ola bilər [9, s.12].

2. Mülkiyyət Qrafikləri: Neo4j, Amazon Neptune, JanusGraph-da istifadə olunur. Düyünlər və kənarlar daxili xassələri (açar-dəyər cütləri) ehtiva edə bilər. Sorğu dilləri: Cypher, Gremlin. Bu model proqram tərtibatçıları üçün daha intuitivdir [8, s.18].

Memarlıq nümunəsi: Facebook TAO

Facebook-un TAO (The Associations and Objects) sisteminə keçidi qrafiklərə mühəndislik yanaşmasının əsas nümunəsidir. Əvvəlcə Facebook MySQL + Memcached yığınından istifadə edirdi. Bununla belə, xəbər lentinin yaradılması keş ardıcıllığı mürəkkəbliyinə səbəb olan "fan-out" sorğuları tələb edirdi.

TAO Memarlığı

TAO, oxunan ağır iş yükləri və coğrafi paylama üçün optimallaşdırılmış paylanmış qrafik mağazasıdır [7, s.496]. TAO, qlobal məlumat mərkəzləri arasında məlumatları sinxronlaşdırarkən belə yüksək mövcudluğa və aşağı gecikməyə imkan verən nəticədə ardıcıl bir model tətbiq edir [7, s.498]. "Vertex-Centric" proqramlaşdırma modelinin istifadəsi minlərlə əmtəə serverində sosial qrafik alqoritmlərinin kütləvi paralelləşdirilməsinə

imkan verir [7, s.502]. Sorğunun optimallaşdırılması və icra planları SQL mühərrikləri kimi, qrafik verilənlər bazası da sorğu optimallaşdırıcılarından istifadə edir. Cypher kimi dillər deklarativdir (nəyin tapılacağını deyil, necə tapılacağını təsvir edir) [8, s.88].

Xidmətdən asılılıq qrafikləri

Mikroservis arxitekturası qovşaqların xidmətlər və kənarların şəbəkə zəngləri (RPC, REST) olduğu bir qrafiki təmsil edir [13, s.5]. Mikroservis arxitekturasında qarşılıqlı təsirlərin və mikroservis asılılıqlarının qraf-nəzəri təhlili rekursiv çağırış zəifliklərini və potensial sistem uğursuzluqlarını müəyyən etməyə kömək edir [13, s.12].

Təhlükəsizlik sahəsində Sybil hücumları sosial qrafikin qanuni "böyük komponenti" ilə güclü əlaqəsi olmayan saxta düyünlərin sıx qruplarını müəyyən etməklə azaldılır [10, s.272]. SybilGuard kimi təsadüfi gəzintilərə əsaslanan alqoritmlər, düyün keçid ehtimallarını təhlil etməklə bu anomal bölgələrin yüksək dəqiqliklə aşkarlanmasına imkan verir [10, s.275].

Zəng qrafikləri: Paylanmış izləmə alətləri (Jaeger, Zipkin) real vaxt rejimində zəng qrafiklərini qurur. Bu, sorğu axınını vizuallaşdırır və çətinlikləri müəyyənləşdirir.

Səbəb Qrafikləri: AlOps-da avtomatlaşdırılmış diaqnoz üçün istifadə olunur. Korrelyasiya qrafiklərindən fərqli olaraq, onlar hadisələrin bir-birinə yönəldilmiş təsirini modelləşdirir, simptomlar və əsas səbəblər arasında fərq qoymağa imkan verir.

Təhlükəsizlik və sybil hücumları

Qrafik analiz təhlükəsizlikdə əsas rol oynayır. Sybil hücumları (təcavüzkar çoxlu saxta hesablar yaradır) topologiya analizi vasitəsilə aşkar edilir. Saxta qovşaqlar tez-tez bir-biri ilə sıx klasterlər əmələ gətirir, lakin qanuni istifadəçilərin "nəhəng komponenti" ilə bir neçə əlaqəyə malikdir [10, s.268]. Təsadüfi gəzintilərə əsaslanan alqoritmlər bu cür anormal bölgələri yüksək dəqiqliklə aşkar etməyə imkan verir [11, s.320].

Ümumi proqramlaşdırma və API dizaynı

Universal qrafik kitabxanalarının yaradılması (C++-da Boost Graph Library kimi) qabaqcıl dil xüsusiyyətləri tələb edir. Şablonlar və Konsepsiyalar (C++): Konseptlər tərtib zamanı tip məhdudiyyətlərini tətbiq edir. ən qısa_yol (Qrafik və g) AdjacencyGraph konsepsiyasını təmin etmək üçün g tələb edə bilər. Bu, alqoritmə virtual funksiya yükü olmadan matrislər, siyahılar və ya gizli qrafiklərlə işləməyə imkan verir [1, s.120].

Sadə tip yoxlamasından başqa, müasir API dizaynı qrafikin topologiyasını onun daxilində saxlanılan məlumatlardan ayırmaq üçün "Ümumi Proqramlaşdırma"dan istifadə edir [1, s.122]. Xüsusiyyət xəritələrindən istifadə etməklə, tərtibatçılar əsas qrafik strukturunu dəyişdirmədən təpə nöqtələrinə çəkilər, rənglər və ya məsafələr əlavə edə bilərlər ki, bu da yaddaşa səmərəli çoxsəviyyəli tətbiqlər üçün vacibdir [5, s.72]. Rust və müasir C++ kimi dillərdə "Məlumatlara Yönləndirilmiş Dizayn" paradiqması "Strukturlar Massivi" (AoS) əvəzinə "Massivlərin Strukturu"nun (SoA) istifadəsini təşviq etməklə bunu daha da irəli aparır. Bu, axtarış zamanı CPU keşinin yalnız cari keçid mərhələsi ilə əlaqəli məlumatları ehtiva etməsini təmin edir və istifadə olunmamış obyekt metaməlumatlarından "keş çirklənməsini" effektiv şəkildə aradan qaldırır [5, s.85]. Bundan əlavə, Facebook-un TAO kimi paylanmış mühitlər üçün API-lərin dizaynı şəbəkə gecikməsini idarə etmək üçün asinxron, bloklamayan çağırışlara keçid tələb edir [7, s.499]. Kənar axtarışının yaddaş dövrü olduğu yerli kitabxanalardan fərqli olaraq, GDBMS API çağırışı çarpaz məlumat mərkəzi keçidini əhatə edə bilər. Robinsonun qeyd etdiyi kimi, yaxşı hazırlanmış Qrafik API, əsas mühərrikin icra planını optimallaşdırmasına və inkişafetdiricinin əl ilə müdaxiləsi olmadan ən səmərəli keçid yolunu təyin etməsinə imkan verən Cypher və ya Gremlin kimi deklarativ sorğu interfeysi təmin etməlidir [8, s.88]. Nəhayət, təhlükəsizlik baxımından vacib sistemlər üçün rəsmi API spesifikasiyaları qrafik mutasiyalarının struktur bütövlüyünü qorumasını təmin edir. Bu, xüsusilə mikroservis asılılıq qrafiklərində aktualdır, burada API kaskad sistem nasazlıqlarına səbəb ola biləcək dairəvi asılılıqların qarşısını almalıdır [13, s.18]. Kompilyasiya vaxtı təhlükəsizliyini aparat təminatına əsaslanan məlumat düzülüşləri ilə birləşdirərək, müasir qrafik API-ləri inkişafetdiricilərə həm yüksək dərəcədə

mücərrəd, həm də fiziki cəhətdən optimal olan sistemlər qurmağa imkan verir [12, s.25].

Nəticə

Tətbiqi proqram təminatı mühəndisliyi kontekstində qrafik alqoritmlərinin tədqiqi və inkişafı bütün texnologiya dəstini əhatə edən bir sahədir: aşağı səviyyəli yaddaş və CPU keşinin optimallaşdırılmasından [5, s.90] planetar miqyaslı paylanmış sistem arxitekturasına qədər. Mühəndislər alqoritmlərin nəzəri saflığı ilə fiziki avadanlıqların sərt məhdudiyyətləri arasında daim tarazlıq saxlamalıdırlar. Hər kənar üçün zaman ölçüsünü özündə birləşdirən zaman qrafikləri sosial qarşılıqlı təsirlərin və maliyyə fırıldaqçılığının təkamülünü təhlil etmək üçün həyati əhəmiyyət kəsb edir [11, s.310]. Bilik Qrafiklərinin və Böyük Dil Modellərinin (BLM) konvergensiyası daha faktiki əsaslandırılmış və izah edilə bilən süni intellekt sistemləri üçün yol açır [12, s.45]. FPGA və GPU-larda aparatla sürətləndirilmiş qrafik emalı, enerji səmərəliliyi baxımından ümumi təyinatlı CPU-ları əhəmiyyətli dərəcədə üstələyən xüsusi boru kəməri arxitekturalarına imkan verir [6, s.14]. Hamiltonun təklif etdiyi kimi, sahənin gələcəyi Qrafik Təqdimatı Öyrənməsindədir, burada GNN-lər mürəkkəb relyasiya məlumatları üçün xüsusiyyət mühəndisliyi prosesini avtomatlaşdırır [12, s.110].

Sadə Genişlik-Birinci Axtarışdan mürəkkəb iyerarxik naviqasiya metodlarına (CH) və paylanmış qrafik verilənlər bazalarına (TAO) təkamül sənayenin riyazi modelləri biznes tələblərinə necə uyğunlaşdırdığını nümayiş etdirir. Gələcəkdə biz qrafik alqoritmlərinin Machine Learning (Qrafik Neyron Şəbəkələri) [12, s.105] ilə daha da yaxınlaşmasını və qrafikin işlənməsini sürətləndirmək üçün xüsusi avadanlıqların (FPGA, GPU) inkişafını gözləyə bilərik. Müasir proqram təminatı memarı üçün qrafik nəzəriyyəsini və onun tətbiqi nüanslarını başa düşmək artıq istəyə bağlı deyil, miqyaslana bilən və səmərəli sistemlər yaratmaq üçün zəruri olan əsas bacarıqdır [12, s.20].

ƏDƏBİYYAT SİYAHISI:

1. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. Introduction to Algorithms (4th ed.). MIT Press. 2022.
2. Dijkstra E.W. A note on two problems in connexion with graphs. Numerische Mathematik, 1959;1(1):269–271.
3. Hart P.E., Nilsson N.J., Raphael B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions on Systems Science and Cybernetics. 1968.
4. Buluc A., Gilbert J.R. The Combinatorial BLAS: Design, implementation, and applications. The International Journal of High Performance Computing Applications. 2011.
5. Drepper U. What Every Programmer Should Know About Memory. Red Hat, Inc. 2007.
6. Georganas E., et al. Communication-Avoiding Sparse Matrix-Vector Multiplication on GPUs. IEEE International Parallel and Distributed Processing Symposium. 2014.
7. Bronson N., et al. TAO: Facebook's Distributed Data Store for the Social Graph. USENIX Annual Technical Conference. 2013.
8. Robinson I., Webber J., Eifrem E. Graph Databases. O'Reilly Media.. 2015.
9. Harris S., Seaborne A. SPARQL 1.1 Query Language. W3C Recommendation. 2013.
10. Yu H., Gibbons P.B., Kaminsky M., Xiao F. SybilGuard: Defending Against Sybil Attacks via Social Networks. ACM SIGCOMM. 2006.
11. Newman M.E.J. Networks. Oxford University Press. 2018.
12. Hamilton W.L. Graph Representation Learning. Morgan & Claypool Publishers. 2020.
13. Dragoni N., et al. Microservices: Yesterday, Today, and Tomorrow. Present and Ulterior Software Engineering. 2017.

Lamiye Rafiq MEMMEDZADE
Master's student at Scool of High technologies and innovative engineering,
Western Caspian University

RESEARCH AND DEVELOPMENT OF GRAPH ALGORITHMS: PERSPECTIVES IN APPLIED SOFTWARE ENGINEERING

Summary

In this research work, the process of transforming graph theory from an abstract mathematical model into a fundamental tool of modern software engineering is analyzed. Practical software engineering must address low-level hardware constraints, including memory management, CPU cache locality, and network latency. The efficiency of graph processing is heavily dictated by data organization within RAM; specifically, this paper highlights Compressed Sparse Row (CSR) as a superior high-performance alternative for static graph representations. The study tracks the development of pathfinding algorithms, contrasting Dijkstra's method with the heuristic-driven efficiency of the A* algorithm. As relational databases struggle with deeply interconnected datasets, the rise of Graph Database Management Systems (GDBMS) becomes inevitable. Using Facebook's TAO as a case study, the paper demonstrates the management of billions of objects and associations. Furthermore, security applications such as Random Walk algorithms for detecting Sybil attacks through topological anomaly analysis are discussed Also, the role of" Random Walk " algorithms in preventing Sybil attacks (fake accounts) and the future prospects of graph neural networks (GNN), as well as the importance of FPGA/GPU-based hardware accelerators, are highlighted.

Keywords: graph theory, computational algorithms, data structures, graph database management systems (GDBMS), system reliability.

Ламие Рафик МАММАДЗАДЕ
Магистрант в Школе Высокие технологии и инновационные инженерные решения,
Западно-Каспийского Университета

ИССЛЕДОВАНИЯ И РАЗРАБОТКИ АЛГОРИТМОВ ДЛЯ РАБОТЫ С ГРАФАМИ: ПЕРСПЕКТИВЫ В ПРИКЛАДНОЙ РАЗРАБОТКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Резюме

В данной исследовательской работе анализируется процесс превращения теории графов из абстрактной математической модели в фундаментальный инструмент современной разработки программного обеспечения. Практическая разработка систем требует решения аппаратных ограничений, таких как управление памятью, локальность кэша процессора и сетевые задержки. Эффективность обработки графов напрямую зависит от организации данных в оперативной памяти (RAM); в частности, рассматривается метод сжатого хранения строкой (Compressed Sparse Row, CSR) как высокопроизводительная альтернатива для статических графов. В работе прослеживается эволюция алгоритмов поиска кратчайшего пути от классического метода Дейкстры до эвристически оптимизированного алгоритма A*. Обосновывается переход от реляционных баз данных к графовым СУБД (GDBMS) в условиях работы со сложными взаимосвязанными данными. На примере архитектуры Facebook TAO демонстрируются механизмы управления миллиардами объектов и ассоциаций. Также анализируются методы обеспечения безопасности, в частности алгоритмы случайных блужданий (Random Walk) для обнаружения атак Сивиллы (фейковых аккаунтов). Также подчеркивается роль алгоритмов" случайной прогулки " в предотвращении атак Sybil

(поддельных учетных записей) и будущие перспективы нейронных сетей графа (GNN), а также важность аппаратных ускорителей на основе FPGA/GPU.

Ключевые слова: теория графов, вычислительные алгоритмы, структуры данных, графовые системы управления базами данных (GDBMS), надежность систем.

Daxil olub: 30.01.2026