Vypracované otázky k MSZ pro rok 2022

Specializace NNET

5. května 2022

Vladimír Dušek, xdusek27

Specializace Počítačové sítě – NNET

1. Architektura superskalárních procesorů a algoritmy zpracování instrukcí mimo pořadí, predikce skoků.
2. Paměťová konzistence a předbíhání operací čtení a zápisu, podpora virtuálního adresového prostoru.
3. Datový paralelismus SIMD, HW implementace a SW podpora.
4. Architektury se sdílenou pamětí UMA a NUMA, zajištění lokality dat.
5. Problém koherence pamětí cache na systémech se sdílenou pamětí, protokol MSI.
6. Paralelní zpracování v OpenMP: Smyčky, sekce a tasky a synchronizační prostředky.
7. Pravděpodobnost, podmíněná pravděpodobnost, nezávislost.
8. Náhodná proměnná, typy náhodné proměnná, funkční a číselné charakteristiky, významná rozdělení pravděpodobnosti.
9. Bodové a intervalové odhady parametrů, testování hypotéz o parametrech.
10. Vícevýběrové testy, testy o rozdělení, testy dobré shody.
11. Regresní analýza.
12. Markovské řetězce a základní techniky pro jejich analýzu.
13. Randomizované algoritmy (Monte Carlo a Las Vegas algoritmy).
14. Problém generalizace strojového učení a přístup k jeho řešení (trénovací, validační a testovací sada, regularizace, předtrénování, multi-task learning, augmentace dat, dropout, ...)
15. Generativní modely a diskriminativní přístup ke klasifikaci (gaussovský klasifikátor, logistická regrese, ...)
16. Neuronové sítě a jejich trénování (metoda gradientního sestupu, účelová (loss) funkce, výpočetní graf, aktivační funkce, zápis pomocí maticového násobení, ...)
17. Neuronové sítě pro strukturovaná data (konvoluční a rekurentní sítě, motivace, základní vlastnosti, použití)
18. Prohledávání stavového prostoru (informované a neinformované metody, lokální prohledávání, prohledávání v nejistém prostředí, hraní her, CSP úlohy)
19. Klasifikace formálních jazyků (Chomského hierarchie), vlastnosti formálních jazyků a jejich rozhodnutelnost.
20. Konečné automaty (jazyky přijímané KA, varianty KA, minimalizace KA, Mihill-Nerodova věta).
21. Regulární množiny, regulární výrazy a rovnice nad regulárními výrazy.
22. Zásobníkové automaty (jazyky přijímané ZA, varianty ZA).
23. Turingovy stroje (jazyky přijímané TS, varianty TS, lineárně omezené automaty, vyčíslitelné funkce).
24. Nerozhodnutelnost (problém zastavení TS, princip diagonalizace a redukce, Postův korespondenční problém).
25. Časová a paměťová složitost (třídy složitosti, úplnost, SAT problém).
26. Postrelační a rozšířené relační databáze (objektový a objektově relační databázový model – struktura a operace; podpora práce s XML a JSON dokumenty v databázích).
27. NoSQL databáze (porovnání relačních a NoSQL; CAP věta a ACID/BASE principy; typy NoSQL databází; dotazování v NoSQL databázích; agregace dat pomocí Map-Reduce a agregační pipeline).
28. Získávání znalostí z dat (pojem znalost; typické zdroje dat; základní úlohy získávání znalostí; analytické projekty a proces získávání znalostí z dat).

29. Porozumění datům (důvod a cíl; popisné charakteristiky dat a vizualizační techniky; korelační analýza).
30. Prostorové DB (problematika mapování prostoru, ukládání, indexace; využití).
31. Indexace (nejen) v prostorových DB (kD-Tree a Grid File (a jejich varianty), R-Tree).
32. Lambda kalkul (definice všech pojmů, operací...).
33. Práce v lambda kalkulu (demonstrace reprezentace čísel a pravdivostních hodnot a operací nad nimi).
34. Haskell – lazy evaluation (typy v jazyce včetně akcí, uživatelské typy, význam typových tříd, demonstrace lazy evaluation).
35. Prolog – způsob vyhodnocení (základní princip, unifikace, chování vestavěných predikátů, operátor řezu – vhodné a nevhodné užití).
36. Prolog – změna DB/programu za běhu (demonstrace na prohledávání stavového prostoru, práce se seznamy).
37. Model PRAM, suma prefixů a její aplikace.
38. Distribuované a paralelní algoritmy – algoritmy nad seznamy, stromy a grafy.
39. Interakce mezi procesy a typické problémy paralelismu (synchronizační a komunikační mechanismy).
40. Distribuované a paralelní algoritmy – předávání zpráv a knihovny pro paralelní zpracování (MPI).
41. Distribuovaný broadcast, synchronizace v distribuovaných systémech.
42. Klasifikace a vlastnosti paralelních a distribuovaných architektur, základní typy jejich topologií.
43. Distribuované a paralelní algoritmy – algoritmy řazení, select, algoritmy vyhledávání.
44. Bezdrátové lokální sítě (Wifi, Bluetooth).
45. Hledání minimální kostry obyčejného grafu (pojmy, stromy a kostry, Kruskalův algoritmus, Primův algoritmus).
46. Hledání nejkratších cest ze zdrojového uzlu do všech ostatních uzlů grafu (Bellman-Fordův algoritmus, Dijkstrův algoritmus).
47. Klasifikace algoritmů volby koordinátora, algoritmus Bully a jeho složitost.
48. Podmínky konsistentního globálního stavu distribuovaného systému.
49. Principy distribuovaného zpracování MapReduce, průběh a jednotlivé operace distribuovaného výpočtu pomocí MapReduce, jeho implementace v Apache Hadoop a Apache Spark.
50. Symetrická kryptografie. Vlastnosti, vlastnosti bezpečného algoritmu, délka klíče, útok silou, příklady symetrických algoritmů, Feistelovy šifry, DES, režimy činnosti, proudové šifry.
51. Asymetrická kryptografie, vlastnosti, způsoby použití, poskytované bezpečnostní funkce, elektronický podpis a jeho vlastnosti, hybridní kryptografie, algoritmus RSA, generování klíčů, šifrování, dešifrování.
52. Hašovací funkce, klíčovaný haš a MAC a jejich použití a vlastnosti.
53. Správa klíčů v asymetrické kryptografii (certifikáty X.509).
54. Základní architektury přepínačů, algoritmy pro plánování, řešení blokování, vícestupňové přepínací sítě.
55. Základní funkce směrovače, zpracování paketů ve směrovači, typy přepínání a architektur.
56. Metody pro výpočet směrování v sítích (Bellman-Ford, Dijkstra, Path vector, DUAL).
57. Řízení toku dat (flow-control) a prevence zahlcení (congestion-control) na transportní vrstvě (MP-TCP, QUIC, SCTP, DCCP).
58. Metody detekce síťových incidentů (signatury, statistické metody) a nástroje (IDS/IPS).
59. Sítě Peer-to-Peer: vlastnosti, chování, způsoby směrování. Strukturované a nestrukturované sítě.

60. Události v JavaScriptu (smyčka událostí, asynchronní programování, klientské události, obsluha událostí)
61. Přenos a distribuce webových dat (URI, protokol HTTP, proudy HTTP, CDN, XHR)
62. Bezpečnost webových aplikací (SOP, XSS, CSRF, bezpečnostní hlavičky HTTP)

Obsah

Kapitola 1

PDS – Sítě Peer-to-Peer: vlastnosti, chování, způsoby směrování. Strukturované a nestrukturované sítě.

1.1 Zdroje

- `08-p2p.pdf`
- `PDS_2021-04-09.mp4`

1.2 Úvod a kontext

[[**todo**]]

1.3 Architektura peer-to-peer sítí

- Peer-to-Peer (P2P) je alternativní architektura vůči client-server.
- Uzly (uživatelé) spolu komunikují napřímo. Každý uzel v síti má stejnou roli. Není zde žádný centrální bod, žádný uzel není nadřazený ostatním[1].
- Jiný způsob adresování – adresování obsahem.
- Jiný způsob směrování – lokální rozhodování, specifická struktura sítí.
- Příklad: BitTorrent, Napster, Gnutella, Skype (dříve), Bitcoin, Bluetooth, instant messaging služby, ...

Logická síť' Základem každé P2P sítě je tzv. logická síť' (*overlay*), která je vystavěna na aplikační vrstvě TCP/IP. Tedy P2P sítě staví nad exitující síť'ovou infrastrukturou. Logická síť' definuje způsob propojení uzlů, směrování, vyhledávání informací, ...

Definice P2P sítě Dynamický soubor nezávislých uzlů (peers), které jsou propojeny a jejichž zdroje (objekty) jsou k dispozici ostatním uzlům v této síti. Zdroje: výpočetní výkon, disková kapacita (soubory), zařízení (tiskárny). Sdílené zdroje jsou přímo přístupné všem uzlům, ty je nabízejí a zároveň využívají. Síť' obsahuje prostředky pro připojení uzlu k síti, hledávání a využití zdrojů, ...

[1] Jsou výjimky.

Typy P2P sítí

- Pravé (pure) – Odebrání libovolného uzlu ze sítě nemá vliv na ztrátu schopnosti sítě poskytovat služby (např. Bitcoin).
- Hybridní – Pro svou činnost využívají centrální uzel pro poskytování části nabízených síť'ových služeb. Centrální bod slouží k autentizaci, indexování, inicializaci uzlu, apod.

Vlastnosti P2P sítí

- Samo-organizovatelnost:
  - Když se uzel připojí nebo odpojí, tak se síť' přeskupí a funguje dále.
  - Decentralizovaná topologie, kde uzly spolupracují na jejím vytvoření a udržování.
  - Každý uzel je zodpovědný za svůj lokální stav a část informací (zdrojů).
  - Uzly mají částečný pohled na topologii sítě (znají své nejbližší sousedy).
- Autonomní chování (samořiditelnost):
  - Uzly se chovají dle svého nejlepšího rozhodování (uzel pouze konzumuje zdroje, ale nechce poskytovat).
  - Rozhodování je lokální a nepredikovatelné $\Rightarrow$ má vliv na topologii sítě, směrování, rozmístění objektů.
  - Uzly se mohou chovat zlomyslně.
  - Problém s ověřování identity uzlů a důvěryhodností (decentralizované řízení).
- Spolehlivost:
  - Spolehlivost sítě roste s redundancí uzlů a informací.
  - Redundance objektů, kopie jsou umístěny ve více uzlech.
- Životnost uzlu:
  - Doba životnosti uzlu je neodhadnutelná $\Rightarrow$ problém s garancí služby.
  - Závisí na subjektivním lokální rozhodnutí.

	Klient – server	Peer-to-Peer	*Výhody/nevýhody P2P*
Směr provozu	Asymetrický	Symetrický	*vs. xDSL, kabelový modem*
Topologie sítě	Stabilní	Dynamická	*Problém spolehlivosti*
Robustnost	Centrální bod	Distribuce zdrojů	*Kritický počet účastníků*
Rozšiřitelnost	Náročné	Součást návrhu	*Neomezený růst sítě*
Bezpečnost	Velký důraz	Problematické	*Chybí odpovědná autorita*
Správa a řízení	Centralizovaný model	Každý uživatel spravuje vlastní uzel	*Samo-organizovaná síť*
Poskytované zdroje	Omezené možnosti	Dynamicky rostoucí počet zdrojů	*Sdílení výpočetní prostoru, paměti, apod.*
Kvalita služeb	Garantovaná	Nelze zajistit	*Dynamicky se měnící*

Obrázek 1.1: Srovnání vlastností P2P a klient-server architektur.

Referenční model P2P

- Mějme množinu uzlů P, množinu zdrojů R a množinu identifikátorů I.
- Struktura logické sítě:
  - mapování zdrojů: $F_R : R \to I$,
  - mapování uzlů: $F_P : P \to I$.
  - Množina uzlů P zpřístupňuje zdroje R v rámci jmenného prostoru I pomocí F_R a F_P.
- Decentralizovaná správa jmenného prostoru:
  - $M : I \to 2^P$
  - Příklad: uzel potřebuje zjistit, kdo má konkrétní zdroj.
- Metrika blízkosti:
  - $d : I \times I \to \mathbb{R}$
  - Příklad: uzel chce konkrétní zdroj, vyhledá všechny uzly, které ho poskytují a na základě metriky blízkosti vybere nejbližšího.

Geometrie sítě a směrování

- Geometrie (topologie).
  - Dynamická, uzly se připojují a odpojují.
  - Množina uzlů P zpřístupňuje zdroje R v rámci jmenného prostoru I pomocí F_R a F_P.
- Směrování.
  - Každý uzel zná své sousedy (uzly a hrany k nim) pomocí relace sousedství: $N : P \to 2^P$ (uzel $\to$ jeho sousedi).

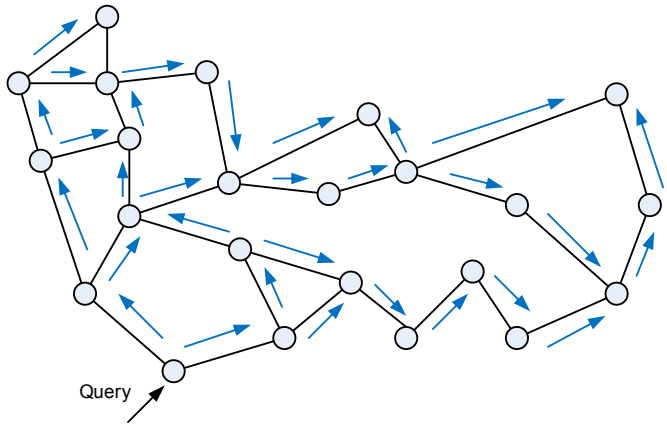
- Mějme předávání zprávy $route(p, m, i)$, kde hledáme cestu pro zprávu m, k uzlu p, který spravuje zdroj i. Směrování: kterému sousedovi mám zprávu předat?
- Distribuovaný proces nalezení cesty v síti P2P na základě lokálních znalostí.
- Směrovací funkce: $R : P \times I \rightarrow 2^P$.
- Směrovací tabulka: každý uzel má svoji, kterou si postupně plní a aktualizuje.

1.4 Nestrukturované sítě

- Neexistuje struktura uložení informace (zdroj (objekt) je umístěn na náhodném uzlu).
- Uzel si vyměňuje zprávy se svými sousedy (dotaz na vyhledávání konkrétního objektu).
- Když uzel hledá objekt, neví, jestli se přibližuje, dostává odpovědi pouze ano-ne.
- Jak v takovém systému směrovat (jak najít objekt skrze identifikátor)?

1.4.1 Záplava (*flooding*)

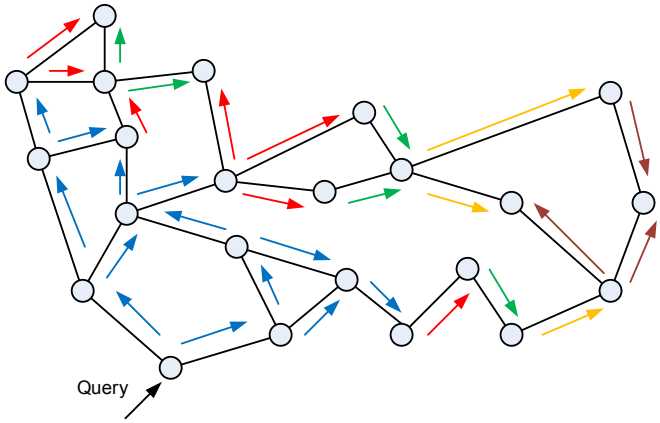
- Uzel pošle dotaz na objekt všem svým sousedům.
  - Pokud soused má objekt, pošle zpět odpověď.
  - Pokud nemá, pošle zprávu svým sousedům.
- Lze omezit pomocí TTL ve zprávě (zamezuje zacyklení a zahlcení sítě).



Obrázek 1.2: Záplava.

1.4.2 Rozšiřující se kruh (*expanding ring*)

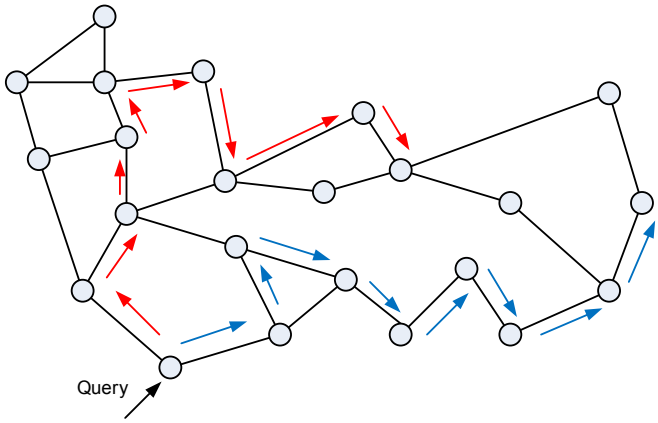
- Uzel pošle dotaz na objekt všem svým sousedům s malým TTL.
  - Pokud objekt je nalezen, hledání končí.
  - Pokud objekt není nalezen, zvýší se TTL a dotaz se pošle znovu.
- Redukuje počet zpráv v síti.



Obrázek 1.3: Rozšiřující se kruh.

1.4.3 Náhodný průchod (*random walk*)

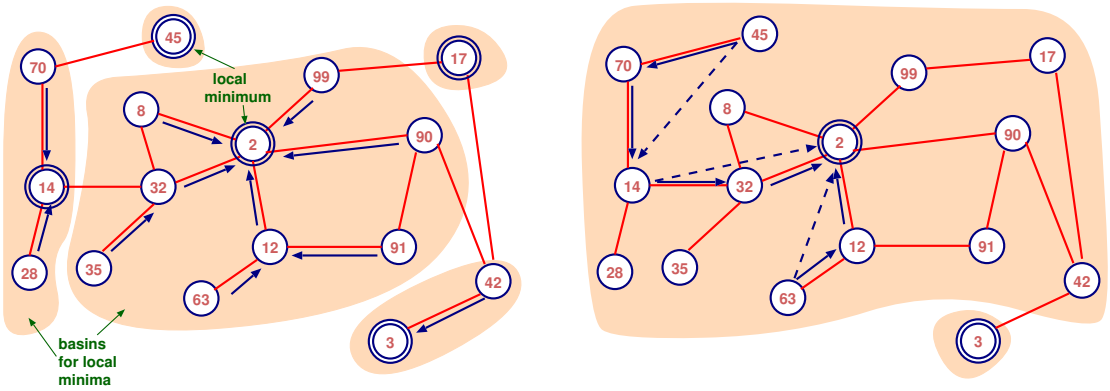
- Uzel pošle dotaz na objekt náhodně vybraným sousedům (i více zároveň).



Obrázek 1.4: Náhodný průchod.

1.4.4 Hledání lokálního minima (*local minimum search*)

- Není pro čistě nestrukturované sítě (přidává lokální vzdálenost).
- Máme množinu uzlů identifikovaných hodnotou x a množinu objektů s identifikátorem w. Úkolem je umístit objekty do sítě uzlů tak, abychom je mohli najít rychle a spolehlivě, tj. jméno uzlu x by mělo být co nejblíže jménu ukládaného objektu w.
- Při hledání používáme metriku vzdálenosti uzlu x od objektu w – $d(x, w)$.
- Uzel u je lokálním minimem pro objekt, pokud je jeho ID nejbližší k ID objektu mezi jeho sousedy do vzdálenosti h kroků.



(a) 1-hop Neighborhoods

(b) 2-hop Neighborhoods

Obrázek 1.5: Příklad hledání lokálního minima. Uzly jsou značeny vzdáleností od objektu. 2 – *hop* – uzel zná sousedy sousedů a zapojuje je do hledání.

1.5 Strukturované sítě

- Kombinují geometrické struktury a směrování.
- Distribuované směrovací algoritmy (metriky: shoda prefixu, euklidovská či lineární vzdálenost, XOR, ...).
- Struktura sítě odpovídá uložení zdrojů.
- Jak směrovat?

1.5.1 Kademlia – Distribuovaná hašovací tabulka (DHT)

- Využívá P2P síť BitTorrent.
- Každý uzel obsahuje informaci o dalších uzlech a souborech.
- Identifikátory uzlů i souborů jsou hash.
- Metrika blízkosti: bitový XOR – $d(a, b) = a \oplus b$.
- Pro směrování používá distribuovanou hašovací tabulku.