Propojovací sítě pro paralelní počítače (taxonomie, požadavky na jejich vlastnosti)

Taxonomie z hlediska toků instrukčních dat:

1. SIMD (Single instruction multiple data) - identické podřízené výpočetní uzly, které jsou řízené jedním tokem makroinstrukcí, které vysílá nadřazený počítač
2. MIMD (Multiple instruction multiple data) - množina samostatných uzlů, z nichž je každý schopen provádět svůj vlastní program ve své vlastní paměti

Taxonomie z hlediska organizace paměti:

1. systémy se sdílenou pamětí
2. systémy s distribuovanou pamětí
3. systémy s distribuovanou sdílenou pamětí (virtuální sdílená paměť)

Taxonomie z hlediska propojovacích sítí:

Typy propojovacích sítí:

1. sdílené komunikační médium (sběrnice)
2. přepínané komunikační médium (uzly připojeny k přepínačům)

Přímé a nepřímé propojovací sítě:

1. přímé sítě: každý přepínač je připojen alespoň k 1 PE
2. nepřímé sítě: některé přepínače jsou připojeny pouze k jiným přepínačům
   I. vícestupňové
   II. stromové

Pravidelné a nepravidelné propojovací sítě:

1. pravidelné sítě: topologie propojení tvoří pravidelný zobecnitelný graf
2. nepravidelné: topologie propojení tvoří náhodný graf

Klasifikace podle doby života propojovacích cest:

1. statické – propojovací cesty zůstávají neměnné
2. dynamické: křížové přepínače (jednoúrovňové, několikaúrovňové), sběrnice

Základní pojmy z teorie grafů

- graf G je určen množinou uzlů a hran G = (V, E).
- stupeň uzlu je počet sousedů.
- graf je k-regulární, když všechny uzly mají stupeň *k*
- kartézský součin grafů:
  - $G = G_1 \times G_2$
  - $V_G = \{(x, y) : x \in V_{G_1}, y \in V_{G_2}\}$
  - $E_G = \{[(x_1, y), (x_2, y)] : [x_1, x_2] \in E_{G_1}\} \cup \{[(x, y_1), (x, y_2)] : [y_1, y_2] \in E_{G_2}\}$
- graf je uzlově symetrický, pokud vypadá z pohledu kteréhokoliv uzlu stejně (pro každou dvojici uzlů existuje automorfismus tak, že se jeden uzel zobrazí na druhý)
- graf je hranově symetrický, pokud vypadá z pohledu kterékoliv hrany stejně (pro každou dvojici hran existuje automorfismus tak, že se jedna hrana zobrazí na druhou)
- excentricita uzlu *u* je vzdálenost uzlu *u* od uzlu, který je mu nejvzdálenější
- průměr grafu je vzdálenost dvou nejvzdálenějších uzlů (maximální excentricita v grafu)
- poloměr grafu je excentricita uzlu s nejmenší excentricitou (ze všech uzlů grafu) (minimální excentricita v grafu)
- souvislost grafu je minimální počet uzlů, jejichž odebrání způsobí rozpojení grafu (obdobně hranová souvislost)
- graf je bipartitní, pokud lze jeho uzly obarvit dvěma barvami tak, aby dva uzly se stejnou barvou nesousedily
- Hamiltonovská kružnice je uzavřená cesta přes všechny uzly
- bisekční šířka grafu je nejmenší počet hran, jejichž odebrání způsobí rozpad grafu na dvě přibližně stejně velké části (obdobně uzlová bisekční šířka)
- chybová vzdálenost uzlů *u* a *v* je maximum z délek všech možných co nejkratších uzlově disjunktních cest mezi *u* a *v*
- chybový průměr je maximum ze všech chybových vzdáleností

Požadavky na vlastnosti propojovacích sítí

1. malý a konstantní stupeň uzlu: technologický požadavek ⇒ řídký graf, malá souvislost a velké vzdálenosti, levné a univerzální směrovače
2. malý průměr a malá průměrná vzdálenost: algoritmický požadavek, snižuje komunikační zpoždění
3. konstantní délka hran: rozmístitelnost uzlů v 3D tak, aby délka kabelů byla konstantní
4. symetrie: zjednodušuje návrh algoritmů (nezáleží na tom, kde výpočet začne ani kterým směrem začne), snazší vnořování a VLSI návrh
5. škálovatelnost:
   - inkrementálně škálovatelná topologie (dostupná pro libovolné N), jinak je částečně škálovatelná
   - je efektivně škálovatelná, pokud přidání *k* uzlů vyžaduje řádově stejně změn, tj. O(k) změn
6. hiearchická rekurzivita (instance nižších dimenzí jsou podgrafy instancí vyšších dimenzí): induktivní návrh a mapování paralelních algoritmů, většinou jen částečná škálovatelnost
7. vysoká souvislost a malé chybové vzdálenosti: důležité z hlediska spolehlivosti a robustnosti, obcházení výpadků nebo přetížených linek, posílání paketů po paralelních cestách
8. velká bisekční šířka: pro metody rozděl a panuj (a jiné rekurzivní problémy) - vysoké přenosové kapacity mezi oběma polovinami (horní mez je N/2)
9. podpora pro směrování a kolektivní komunikační operace
10. existence hamiltonovských cest a 2-barvení: hamiltonovská cesta je vnoření kružnice, zjednodušuje návrh algoritmů, které používají lineární číslování uzlů
11. vnořitelnost: efektivní zobrazení daného grafu procesů do sítě procesorů, schopnost efektivně simulovat jiné topologie

Přímé propojovací sítě

Ortogonální

Binární hyperkrychle Q_n:

- 2^n uzlů, $n \cdot 2^{n-1}$ hran, průměr n, bisekční šířka 2^{n-1}, souvislost n
- regulární (všechny uzly mají stejný stupeň a tedy stejný počet sousedů)
- logaritmický stupeň uzlů ⇒ řídká hyperkubická síť
- hiearchicky rekurzivní: Q_n lze rozložit na dvě hyperkrychle Q_{n-1}
- uzlová a hranová symetrie
- největší možná bisekční šířka ⇒ ideální pro D&C algoritmy

- vyvážený bipartitní (dáno paritou) a Hamiltonovský graf (dán Grayovými kódy)
- simuluje efektivně téměř všechny známé topologie
- vzdálenost uzlů dána počtem rozdílných bitů
- problém alokace na víceuživatelském počítači - podkrychle, problém fragmentace
- nedostatky: nedostatečná škálovatelnost, logaritmický stupeň
- přeložení z uzlu *u* do uzlu *v* (souvisí s uzlovou symetrií): x = x xor (u xor v) (tedy u se zobrazí na v a v se zobrazí na u)
- permutace dimenzí (souvisí s hranovou symetrií): systematická změna pořadí souřadnic všech uzlů
- částečně, ale efektivně škálovatelná

Mřížka M(z_1, z_2, …, z_n)

- $\prod_{i=1}^{n} z_i$ uzlů, $\sum_{i=1}^{n} \left((z_i - 1) \prod_{j=1, j \neq i}^{n} z_j \right)$ hran, průměr = $\Omega(\sqrt[n]{|V|})$, bisekční šířka $\Omega(\frac{\prod_{i=1}^{n} z_i}{\max_i z_i})$ (pokud je maximální z_i sudé, pak platí rovnost bez omega)
- 1D - inkrementálně škálovatelné, protipól úplného grafu
- 2D, 3D - nejpraktičtější pro použití
- není regulární ⇒ není uzlově symetrická
- velký průměr
- hierarchicky rekurzivní
- problém alokace jako u hyperkrychlí
- bipartitní
- Hamiltonovská cesta existuje vždy
- částečně, ale neefektivně škálovatelná
- konstrukce pomocí kartézského součinu (jako u krychlí)

Toroid K(z_1, z_2, …, z_n)

- mřížka, kde každá lineární řada je uzavřena do kružnice
- regulární, uzlově symetrický
- poloviční průměr oproti mřížce, dvojnásobná bisekční šířka
- bipartitní pokud jsou všechny délky stran sudé
- částečně škálovatelný (ještě méně efektivní v porovnání s mřížkami)

Řídké hyberkubické

Jedná se o řídké grafy odvozené od hyperkrychle rozvinutím každého uzlu hyperkrychle do více uzlů.

Kružnice propojené krychlí CCC_n:

- počet uzlů $n \cdot 2^n$, počet hran $n \cdot 2^{n-1} + n \cdot 2^n$, průměr $(2n - 2) + \lfloor n/2 \rfloor$ (pro CCC_3 je průměr 6), stupeň 3, bisekční šířka 2^{n-1}
- je uzlově symetrický, není hranově symetrický (hyperkubické a kružnicové hrany)
- není hieararchicky rekurzivní
- pro sudá *n* je vyvážený bipartitní

Zabalený motýlek wBF_n:

- jako CCC_n, ale hyperkubické hrany nespojují stejnolehlé uzly v sousedních kružnicích, ale uzly sousední (jak vlevo tak vpravo)
- uzel má tedy dva sousedy ve své kružnici a dva sousedy v sousedních kružnicích
- základní vlastnosti má stejné jako CCC_n, až na to, že má více hran ($n \cdot 2^{n+1}$), větší bisekční šířku 2^n a menší průměr $n + \lfloor n/2 \rfloor$

Obyčejný motýlek oBF_n:

- počet uzlů $(n + 1) \cdot 2^n$, počet hran $n \cdot 2^{n+1}$, průměr *2n*, bisekční šířka 2^n
- pozor: uzly o stejné x-ové souřadnici jsou číslovány od nuly, na každé x-ově souřadnici jich je tedy *n + 1* (narozdíl od CCC_n a wBF_n)
- není regulární ani uzlově symetrický
- hieararchicky rekurzivní

Nepřímé vícestupňové sítě

Nepřímé sítě:

1. vícestupňové nepřímé sítě
2. stromové sítě
3. nepravidelné sítě
   I. jednosměrné
   II. obousměrné

Nepřímé vícestupňové sítě:

- existuje jedinečná cesta mezi libovolnou dvojicí vstupu a výstupu
- mají n sloupců tvořených 2^{n-1} přepínači 2×2:
  - levý sloupec je připojen ke vstupům (procesory) hranami nultého stupně
  - pravý sloupec připojen hranami n-tého stupně k výstupu (procesory, paměti)
- druhy permutačních stupňů:
  1. dokonalé promíchání (Perfect Shuffle), σ: rotace vlevo o jeden bit
  2. motýlek (Butterfly), β_i: záměna posledního a *i*-tého bitu
  3. základní (Baseline), δ_i: rotace doprava posledních *i*+1 bitů, bity před tím nechávám beze změny
- příklady:
  1. základní síť (sigma, základní, základní, základní)
  2. motýlek (motýlek, motýlek, motýlek, motýlek)
  3. nepřímá hyperkrychle (sigma, motýlek, motýlek, motýlek)
  4. omega síť (sigma, sigma, sigma, sigma)
- přestavitelné - Benešova (dva motýlci otočení k sobě „zády") - lze realizovat jakoukoliv permutaci bez kolize
- obousměrné: přepínače se umí vevnitř přepojit tak, že jeden vstup jde na druhý vstup

Problém vnoření

Základní pojmy:

- vnoření zdrojového grafu G do cílové sítě H (opět graf) je dvojice zobrazení - z uzlů na uzly, z hran na množinu všech cest

- **zatížení cílového uzlu** - počet zdrojových uzlů na něj namapovaných (značí se load)
- **zatížení vnoření** - maximální zatížení uzlu (ze všech uzlů cílové sítě)
- expanze - poměr velikosti cílové sítě (počet uzlů) a zdrojového grafu (počet procesorů)
- **dilatace zdrojové hrany** - protažení zdrojové hrany v cílové síti (tj. na jak dlouhou cestu byla hrana namapována) (značí se dil)
- **dilatace vnoření** - maximální dilatace (ze všech hran zdrojového grafu)
- **linkové zahlcení cílové linky** - počet zdrojových hran využívajících cílovou hranu (namapovaných na cestu procházejících cílovou hranou) (značí se **ecng**)
- **linkové zahlcení vnoření** - maximum ze všech linkových zahlcení
- **uzlové zahlcení cílového uzlu** - počet cest, na které je namapovaná nějaká zdrojová hrana a procházejí cílovým uzlem (značí se ncng)

Rozdělení:

1. statický
   - známe velikost a strukturu grafu procesů
   - máme počítač s distribuovanou pamětí se známou topologií sítě
   - jak mapovat graf procesů na tento stroj, aby výpočet byl co nejefektivnější?
2. dynamický
   - procesy dynamicky vznikají, neznáme velikost ani strukturu grafu procesů, jen částečné informace (např. max. počet potomků jednoho procesu)
   - máme počítač s distribuovanou pamětí se známou topologií propojovací sítě
   - jak distribuovat dynamicky vznikající procesy mezi procesory tak, aby výpočet byl co nejefektivnější?

Vnoření do hyperkrychle:

- vnoření do hyperkrychle může být popsáno pomocí:
  1. značení uzlů - ohodnocení uzlů binárními adresami
  2. značení hran - ohodnocení hran čísly dimenzí
- cesty a kružnice: pomocí Grayových kódů (Binární zrcadlový Grayův kód)
- v krychli neexistují kružnice liché délky
- úplné binární stromy:
  - CBT_n není podgrafem Q_{n+1}, protože CBT_n není, zatímco Q_{n+1} je, vyvážený bipartitní graf
  - statická varianta - vnořujeme do Q_{n+1}
    - rekurzivní zdvojování kořene
      - load = ecng = 1, dil = 2
    - inorder číslování
      - dil = ecng = 2, load = 1
  - dynamická varianta (pravděpodobnostní algoritmus)
    - kořen je umístěn do libovolného uzlu hyperkrychle
    - podle náhodného rozhodnutí listový proces umístí svého levého syna do stejného uzlu a pravého syna do sousedního nebo opačně
- mřížky: $M(z) \subseteq Q_{\lceil \log z \rceil}$
- toroidy: $K(z) \subseteq Q_{\lceil \log z \rceil}$ pro všechna z_i sudá, jinak se vnořuje s load = 1 a dil = 2
- kružnice propojené krychlí: $CCC_n \subset Q_{n+\lceil \log n \rceil}$ pro *n* sudé, jinak se vnořuje s load = 1 a dil = 2
- obyčejný motýlek: $oBF_n \subset Q_{n+\lceil \log n \rceil}$
- zabalený motýlek: $wBF_n \rightarrow_{emb} Q_{n+\lceil \log n \rceil}$ s dil = O(1) a ecng = O(1)

Vnoření do mřížek a toroidů:

- důležité v praxi - navrhování VLSI obvodů
- hyperkrychle: Peanova křivka (spojuje uzly v lexikografickém pořadí při dělení střídavě podle osy x a osy y)
- čtvercová mřížka do obdélníkové: hadi
- obdélníková mřížka do čtvercové: vyplníme hadem čtverec (dil = 1, load = ecng = 2)
- lineámích polí a kružnic: v toroidu nalezneme Hamiltonovskou kružnici, v mřížce Hamiltonovskou cestu
- mřížka do toroidu: mřížka je podgrafem toroidu
- toroidu do mřížky: load = 1, dil = ecng = 2 (pomocí kartézské dekompozice)

Lineární pole nebo kružnici lze vnořit do jakéhokoli grafu G

1. vytvoř kostru T grafu G
2. prováděj DFS na T a při uzavírání uzlu umisťuj uzly vnořovaného grafu

Simulace sítí a výpočetní ekvivalence sítí

Základní pojmy:

- kvaziizometrické sítě - existují vnoření (oběma směry) s konstantními hodnotami měřítek vnoření
- výpočetně ekvivalentní sítě - dokáží se vzájemně simulovat s konstantním zpomalením (na jeden krok konstantní počet kroků)

Vztahy:

- hyperkrychle simuluje optimálně téměř každou známou propojovací topologii
- mřížky a toroidy jsou výpočetně ekvivalentní
- kružnice propojené krychlí a oba typy motýlků jsou výpočetně ekvivalentní

Směrovací algoritmy a architektura směrovačů

Klasifikace komunikačních problémů:

1. jeden jednomu: žádné problémy se zablokováním či zahlcením
2. jeden mnoha: vysílání ve skupině (MC), vysílání jeden-všem (OAB), rozesílání jeden-všem (OAS)
3. všichni-všem: vysílání (AAB), rozesílání (AAS)

Základní pojmy (architektura):

- výpočetní uzel = počítač + směrovač (poskytuje připojení do propojovací sítě a zajišťuje funkci mezilehlého uzlu)
- směrovač = centrální přepínač + vstupní a výstupní vnitřní (k procesoru) a vnější kanály (propojují směrovače mezi sebou - definují topologii sítě) + jednotky pro směrování a řešení konfliktů
- kanály mohou být jednosměrné, poloduplexní a plně duplexní
- u výpočetního uzlu s distribuovanou pamětí (na sběrnici připojen procesor, paměť atd.) je směrovač připojen na sběrnici
- přepínač = propojuje vstupní kanály na výstupní

Klasifikace směrovacích algoritmů:

1. rozhodování o směrování (kdy a kde je provedeno)
   I. distribuované (inkrementální): rozhodnutí podle cílových adres v hlavičkách zpráv

   II. zdrojové - celou trasu předpočítá zdrojový uzel
   III. hybridní - zdrojový uzel předpočítá mezilehlé uzly, přesné trasy mezi nimi jsou distribuovaně určeny směrovači
   IV. centralizované (u SIMD strojů s centrálním řadičem)
2. adaptivita
   I. deterministické - vždy generují tutéž trasu pro tutéž dvojici zdroj-cíl (e-cube, XY)
   II. datově necitlivé (ke stavu sítě)
   III. adaptivní - vyhýbají se zahlceným nebo porouchaným částem sítě
3. minimálnost
   I. minimální (lačné) - každý krok blíže k cíli, náchylné k zablokování
   II. neminimální - posílání paketů dále od svého cíle je možné, náchylné k dynamickému zablokování
4. progresivnost
   I. progresivní (každé minimální) - každý krok alokuje nový kanál a délka cesty vzroste; při zablokované trase paket buď čeká, nebo je odkloněn (náhodně nebo adaptivně)
   II. s návratem - při zablokování se stáhne zpět a uvolní obsazené kanály
5. implementace
   I. konečný automat (HW nebo SW)
   II. směrovací tabulky
      - intervalové směrování (nemá tak velké paměťové nároky, protože v tabulce je vždy záznam pro interval cílových adres)

Minimální směrovací algoritmy typu jeden jednomu:

- hyperkrychle: e-cube směrování (porovnávají se bity v cílové a aktuální adrese, při neshodě se provede negace příslušného bitu, což odpovídá přechodu do cílové dimenze)
- mřížky: dimenzně uspořádané směrování - XY (pro 2D) nebo XYZ (pro 3D) směrování - podobné e-cube, jenom se porovnávají nebinární čísla (nejprve se dostanu do správné pozice v rámci jedné dimenze, pak druhé a nakonec třetí dimenze)
- toroidy: dimenzně uspořádané směrování, ale o něco složitější (je třeba se rozhodnou, na kterou stranu mám v rámci dané dimenze jít, abych to měl blíž)
- kružnice propojené krychlí:
  - minimální směrování je obtížné
  - neminimální:
    1. zjisti, ve kterých bitech se liší adresa výchozí a cílové kružnice
    2. nechť *i* je pozice prvního takového bitu zleva
    3. přesuň se do i-tého uzlu v rámci výchozí kružnice
    4. pomocí e-cube algortimu přejdi do cílové kružnice
    5. v ní se přesuň do cílového uzlu
- zabalený motýlek: e-cube směrování
- obyčejný motýlek: e-cube směrování (existuje pouze jediná cesta mezi (0, x) a (n, y))

Základní techniky přepínání

Základní pojmy:

- zpráva - jednotka komunikace z hlediska programu, proměnná délka
- paket - pevná délka, obsahuje směrovací informace
- flit - linková vrstva, několik typů (hlavičkové, datové, …)
- fit - fyzická vrstva (přenesena přes jednu fyzickou linku v jednom cyklu)

Metrika:

- rychlost kanálu $q[Bs^{-1}]$ je špičková rychlost přenosu bitů po jednom fyzickém vodiči
- zpoždění kanálu $t_m = \frac{1}{q}[sB^{-1}]$ je zpoždení mezi sousedními směrovači na jeden fit
- startovní zpoždění $t_s[s]$ je SW a HW zpoždění ve zdrojovém a cílovém uzlu nutné pro:
  - zformátování a složení paketu
  - validace dat a jejich kopírování mezi pamětí uzlu a frontou směrovače
- směrovací zpoždění $t_r[s]$ je čas pro směrovací rozhodnutí během budouvání trasy
- přepínací zpoždění $t_w[sB^{-1}]$ je čas přenosu přes přepínač ze vstupních na výstupní kanály
- velikost paketu $\mu[B]$
- délka přenosové trasy δ
- platí: $t_s \gg t_m \approx t_w \approx t_r$
- doba přenosu μ-bytového paketu mezi dvěmi sousedními směrovači je $\mu t_m[s]$

Přepínání kanálů (CS)

Princip:

1. konstrukce propojovací cesty: před vlastním přenosem je vyslána směrovací sonda délky p > 1, která rezervuje fyzické linky
2. když dorazí do cíle, tak je poslán nazpět potvrzovací flit
3. přenos zprávy
4. zrušení spojení (například cílovým uzlem nebo posledními datovými bity)

Vlastnosti:

- po průchodu sondy obvod funguje jako jediný vodič
- neexistují žádná omezení na délku zprávy
- výhodné, pokud jsou zprávy dlouhé a málo časté
- přenos zprávy délky μ na vzdálenost δ trvá čas $t_{CS}(\mu, \delta) = t_s + \delta(t_r + (p+1)(t_w + t_m)) + \mu t_m$, kde:
  - sonda pro cestu do cíle potřebuje čas $\delta(t_r + p(t_w + t_m))$
  - potvrzení putuje zpět čas $\delta(t_w + t_m)$
  - přenos dat trvá čas μt_m
- zjednodušení: $t_{CS}(\mu, \delta) \doteq t_s + \delta t_d + \mu t_m, t_d = t_r + t_w + t_m$

Ulož-pošli-dál (SF)

Vlastnosti:

- zprávy rozděleny do stejně velkých paketů
- přepínání paketů - každý paket je individuálně směrován do cíle
- směrovací rozhodnutí učiněno až po přijetí celého paketu
- fronty musí mít takovou kapacitu, aby se do nich vešel celý paket
- výhodné pro kratké a časté zprávy (z celé trasy je obsazen nejvýše jeden kanál)

- komunikační zpoždění je úměrné součinu velikosti paketu a délky trasy ⇒ potřebujeme minimální směrování a nízký průměr sítě, aby zpoždění zůstalo rozumné
- přenos zprávy délky μ na vzdálenost δ trvá čas $t_{SF}(\mu, \delta) = t_s + \delta(t_r + (t_w + t_m)\mu)$
- zjednodušení: $t_{SF}(\mu, \delta) \doteq t_s + \delta\mu t_m$ ⇒ citlivé na vzdálenost

Průřezové (VCT)

Vlastnosti:

- princip funkce jako u SF, ale směrovací rozhodnutí provedeno a flit přeposlán ihned po přijetí hlavičkového fitu (tj. nečeká se na celý paket)
- každý další flit je uložen a rovněž hned prořízne do dalšího směrovače (je-li výstupní kanál volný)
- fronty přesto musí mít takovou kapacitu, aby se do nich vešel celý paket (kdyby nemohla hlavička pokračovat dál, může ji celý zbytek paketu „dohnat")
- všechny fronty podél trasy jsou pro jiné komunikační požadavky blokovány (protože pouze hlavičkový flit obsahuje směrovací informace)
- z uvedených technik je nejnákladnější a nejsložitější, ale díky vyspělosti technologií se dnes nejvíce používá
- přenos zprávy délky μ na vzdálenost δ trvá čas $t_{VCT}(\mu, \delta) = t_s + \delta(t_r + t_w + t_m) + \mu \max(t_w, t_m)$, kde:
  - zpoždění hlavičky při provádění směrovacích rozhodnutí, přepínání a přesunech je $\delta(t_r + t_w + t_m)$
  - rychlost přenosu řetězce flitů, jakmile dosáhne hlavička cíle a pokud mají směrovače vstupní a výstupní fronty je $\max(t_w, t_m)$ (pokud mají pouze vstupní nebo pouze výstupní, je to $t_w + t_m$)
- zjednodušení: $t_{VCT}(\mu, \delta) \doteq t_s + \delta t_d + \mu t_m, t_d = t_r + t_w + t_m$ ⇒ necitlivé na vzdálenost

Červí (WH)

Vlastnosti:

- princip naprosto stejný jako VCT, ale vyrovnávací paměti mají kapacitu jen pro jeden flit (nebo malý počet flitů)
- náchylné na zablokování - když hlavička nemůže pokračovat dál, zamrzne za ní celý řetěz flitů
- proč se používá: umožňuje malé, levné a rychlé směrovače
- přenos zprávy délky μ na vzdálenost δ trvá čas $t_{VCT}(\mu, \delta) = t_s + \delta(t_r + t_w + t_m) + \mu \max(t_w, t_m)$
- zjednodušení: $t_{WH}(\mu, \delta) \doteq t_s + \delta t_d + \mu t_m, t_d = t_r + t_w + t_m$ ⇒ necitlivé na vzdálenost

Problém zablokování a jeho řešení (graf kanálových závislostí, neblokující směrování, virtuální kanály)

Základní pojmy:

- zablokování: situace, kdy paket v síti již obsadil několik kanálů a pro další postup požaduje kanál, který je již obsazen jiným paketem, ale ten je ve stejné situaci (existuje tedy cyklus)
- řešení:
  1. detekce a zotavení: nejméně opatrné, možný velký zisk, ale i ztráty (např. odebrání kanálu paketu nebo zrušení paketu)
  2. prevence: kanály jsou paketům přidělovány tak, že nemůže dojít k zablokování (de facto přepínání kanálů) ⇒ malé využití prostředků sítě
  3. vyhnutí se zablokování (střední cesta): kanály jsou při postupném budování spojení přidělovány tak, aby výsledný globální stav byl bez zablokování

Ortogonální topologie (hyperkrychle, mřížky)

1. neblokující směrování: seřazení dimenzí (směrů) a jejich přidělovat jen v klesajícím pořadí (tím se vyloučí vznik cyklických žádostí) - typičtí představitelé jsou XY směrování, XYZ směrování a e-cube směrování
2. graf kanálových závislostí Z(G, R) - orientovaný graf
   - jeho uzly jsou kanály sítě a dva uzly jsou propojeny orientovanou hranou právě tehdy, když směrovací funkce R může pro tyto uzly směrovat paket z daného vstupního kanálu na výstupní
   - pokud je Z(G, R) acyklický, nemůže dojít k zablokování
3. virtuální kanál: každý fyzický kanál bude nahrazen dvěma nebo více virtuálními kanály, které se budou fyzicky spravedlivě multiplexovat na bázi flitů (používá se v toroidech)

Nepravidelné topologie

Algoritmy nahoru / dolů:

1. zkonstruujeme kostru s jediným kořenem (např. průchodem do šířky)
2. každý uzel má nějaký identifikátor, kořen bude mít nejnižší ID
3. určíme orientaci (od uzlu dále od kořene do uzlu blíže kořenu, pokud jsou na stejné hladině tak od vyššího ID k nižšímu)
4. dovolené jsou pouze ty cesty, které používají nejprve jen linky nahoru a následně jen linky dolů

Algoritmy pro permutace

- permutační algoritmy říkají, jak nastavit směrovače, aby vytvořená permutace mezi procesory spotřebovala minimální počet kroků a minimální množství paměti
- každý procesor je zdrojem maximálně jedné zprávy
- každý procesor přijímá maximálně jednu zprávu
- pokud se permutace účastní všechny procesory, jedná se o úplnou permutaci
- uvažujeme pouze ortogonální a hyperkubické sítě
- ve většině případů se uvažují SF sítě
- časově optimální permutace: počet kroků je O(průměr sítě)
- paměťově optimální permutace: pomocné fronty směrovačů mají velikost O(1)
- pro minimalizaci paměťových nároků permutačního směrování se používájí metody založené na randomizaci, seřazení paketů nebo s předvýpočtem

Přímé

- 1D mřížka všeportová s duplexními kanály
  - strategie nejvzdálenější nejdřív (nejprve je z uzlu vyslán paket, jehož cíl je nejdále) - každý paket potřebuje k dosažení cíle nejvýšek *n - 1* kroků
  - permutace 1-1: všechny pakety se dají do pohybu v kroku 1 a jejich pohyb je bezkolizní
- kD mřížka M_k(n, n, …) všeportová s duplexními kanály
  - XY směrování (případně jeho varianta pro více dimenzí)
  - přesun nejpreve do sloupců a pak ve sloupcích do řádků vyžaduje k · (n - 1) kroků
  - směrovače musí mít pomocné fronty pro $\max(2k - 2, n - 2 - \frac{n-3}{2k-1})$ paketů, což je obecně problém

Náhodné

- randomizace a náhodné permutace je nejjednodušší postup změnšení maximální velikosti fronty
- rovnoměrné rozptýlení paketů činí jejich shlukování méně pravděpodobné
- první metoda:

1. vygenerujeme náhodně mezilehlý uzel
2. nejprve pošleme paket do mezilehlého uzlu a pak do cílového (použijeme minimální směrování)

- druhá metoda:
  1. každý sloupec rozdělen do $\log n$ intervalů o velikosti $\frac{n}{\log n}$
  2. každý paket je nejprve směrován k náhodně vybranému cíli uvnitř svého intervalu, pak v rámci aktuálního řádku do cílového sloupce a nakonec do cílového řádku
  3. při kolizi se použije strategie nejvzdálenější nejdřív

Založené na třídění

1. Pakety jsou setříděny lexikograficky do globálního hada po sloupcích (dle adres cílových sloupců) - $T_{sort}(M(n, n))$ kroků.
2. Každý druhý sloupec převrácen (*n - 1* kroků). V každém řádku je pak maximálně jeden paket určený pro konkrétní sloupec.
3. Přesun do cílových sloupců (permutace v rámci jednotlivých řádků) - *n - 1* kroků.
4. Přesun do cílových řádků (permutace v rámci jednotlivých sloupců) - *n - 1* kroků.

S předvýpočtem (offline)

1. Pro všechny sloupce se předpočítají takové permutace, aby po jejich provedení byl v každém řádku nejvýše jeden paket určený pro jakýkoliv daný sloupec (lze využít Hallovu větu o párování).
2. Ve sloupcích se provede permutace - *n - 1* kroků.
3. Přesun do cílových sloupců - *n - 1* kroků.
4. Přesun do cílových řádků - *n - 1* kroků.

Spodní meze na počty kroků a časová zpoždění a efektivní a optimální algoritmy

Rozdělení komunikačních modelů:

- počet paralelně použitelných portů: 1-portové, k-portové nebo všeportové
- směrovost kanálů: simplexní, poloduplexní a plně-duplexní
- velikost paketu: v kombinujícím modelu se bude velikost paketu μ měnit
- přepínání: pomalý SF nebo rychlý WH
- kroky: spodní mez ρ, horní mez *r*
- čas: spodní mez τ, horní mez *t*

Všeportová plně duplexní nekombinující hyperkrychle Q_n:

- OAB: $\rho = n$ (průměr sítě)
- AAB: $\rho = \lceil \frac{2^n - 1}{n} \rceil$ (každý uzel má obdržet $2^n - 1$ paketů a přitom může obdržet nejvýše *n* v jednom kroku)
- OAS: $\rho = \lceil \frac{2^n - 1}{n} \rceil$ (zdroj má vyslat celkem $2^n - 1$ paketů a přitom v jednom kroku jich může vyslat nejvýše *n*)
- AAS: $\rho = 2^{n-1}$

OAB

SF sítě:

- nemá smysl uvažovat kombinování
- výstupně všeportová síť:
  - ρ = průměr sítě
  - záplavový algoritmus - když přijmu paket poprvé, zkopíruju si ho a pošlu všem zbývajícím sousedům
  - aby nedocházelo k duplikacím, lze použít kostru vytvořenou např. průchodem do šířky
- 1-portový model:
  - $\rho = \max(diam(G), \log |V(G)|)$, protože v jednom kroku se může počet informovaných uzlů maximálně zdvojnásobit
  - obecně spodní mez v nesymetrické d-portové síti: $\rho = \max(exc(s, G), \log_{d+1} |V(G)|)$
  - obecně spodní mez v uzlově symetrické d-portové síti: $\rho = \max(diam(G), \log_{d+1} |V(G)|)$
- EREW PRAM: binární zdvojování (obrácená paralelní binární redukce)
- úplný graf a hyperkrychle: binomiální kostra (SBT_n se skládá ze dvou SBT_{n-1} s propojenými kořeny) a na ní buď záplavový algortimus (všeportová síť) nebo binární zdvojování (1-portová síť)
- mřížky: dimenzionálně uspořádané kostry (zobecnění SBT) - když přišel paket z dimenze *i*, tak ho pošli dál v této dimenzi a rozešli ho všem sousedům v dimenzích větších než *i* (na obě strany)

WH sítě:

- komunikace necitlivá na vzdálenost (takže se u spodní meze neuplatní průměr)
- výstupně všeportová síť: $\rho = \log |V(G)|$
- d-portový model: $\rho = \lceil \log_{d+1} |V(G)| \rceil$
- hyperkrychle:
  - 1-portová - binomiální kostra jako u SF
  - všeportová
    - princip algortimu: nejprve se pošle paket do opačného uzlu a začnou se budovat dvě částečné binomiální kostry proti sobě
    - $\rho = \lceil \frac{n}{\log(n+1)} \rceil$
    - algortimus Ho-Kao - rozdělíme krychli na podkrychle dimenze ≤ 6 (tam je výše popsaný postup optimální) a na ně algoritmus aplikujeme
- 1-portová mřížka: binární zdvojování (tedy simulace binomiální kostry na SF hyperkrychli)
- 1-portový toroid: binární zdvojování (díky uzlové symetrii jednodušší)
- všeportová mřížka a toroid:
  - $\rho = \log_{2n+1} N$, kde *N* je počet uzlů
  - 3-fázový diagonální algoritmus (zobecněná diagonála)
    1. zdrojový uzel doručí paket do každého řádku (logaritmický počet kroků - horizontální pásy)
    2. seřazení paketů na hlavní diagonálu
    3. diagonální uzly informují zbývající uzly postupným dělením do diagonálních pásů - tedy opět logaritmicky

Vysílání skupině (MC)

- jeden uzel informuje pouze podmnožinu uzlů *M*
- pro SF lze použít modifikace OAB
- pro WH obtížnější (klasický OAB by byl velmi neefektivní)
- d-portová WH síť: $\rho = \lceil \log_{d+1}(|M| + 1) \rceil$
- 1-portové WH sítě s plně duplexními kanály:

- hyperkrychle: lexikografické setřídění, binární zdvojování
- mřížka:
  1. rozdělit lexikograficky setříděnou posloupnost do dvou polovin (horní a dolní)
  2. je-li zdroj ve spodní polovině, poslat paket prvnímu uzlu v horní polovině, jinak poslednímu uzlu spodní poloviny
  3. opakuj rekurzivně na obě poloviny

OAS

- zdrojový uzel pošle individuální paket každému uzlu, tedy na počátku má zdroj N-1 paketů velikosti μ, na konci získá každý uzel 1 paket velikosti μ
- podobná složitost jako AAB, neboť v AAB má každý uzel obdržet N - 1 paketů (v OAS zdroj vyslat)
- nekombinující model
  - zdroj posílá všech N - 1 paketů jako samostatné jednotky:
    - 1-portová SF síť: hamiltonovská cesta + FF
    - 1-portová WH síť: cíle v libovolném pořadí
  - výstupně všeportové sítě
    - $\rho = \lceil \frac{V(G)-1}{d} \rceil$, kde *d* je stupeň zdrojového uzlu
    - kostra se bude skládat z *d* podstromů
- kombinující model
  - jako OAB až na to, že velikost zprávy postupně klesá (ρ je stejné jako u OAB)
  - SF mřížky a toroidy - postupné předávání mezi sousedy (pipelining)
  - WH mřížky a toroidy - binární zdvojování
  - hyperkrychle
    - 1-portová: SF binomiální kostra, kde velikost zprávy klesá na polovinu
    - více-portová: OAS varianta dvojitých stromů (Ho-Kao)

AAB

SF kombinující sítě:

- d-portová síť: $\rho = diam(G)$
- všeportový model
  - plně duplexní: záplavový algoritmus (zkombinovat pakety ze všech směrů, odstranit duplikáty a rozeslat)
  - poloduplexní:
    - metoda soustřeď-rozešli - soustřeď všechno jednomu (AOG - opak k OAS) a ten akumulovaný paket rozešle všem (OAB)
    - bipartitní graf - střídavé přenášení zleva doprava a zprava doleva (funguje jen pro bipartitní sítě)
- 1-portový plně duplexní model:
  - je možné použít metodu soustřeď-rozešli
  - 1D mřížky a toroidy: střídavě licho-suché a sudo-liché výměny (výměna informací nejprve s levým sousedem, pak s pravým)
  - vícerozměrné mřížky a toroidy: rozklad po dimenzích a pak opět licho-sudé a sudo-liché výměny (v rámci jednotlivých dimenzí)
  - hyperkrychle: licho-sudé a sudo-liché výměny (v každé dimenzi jedna výměna, tj. $\rho = n$), velikost zprávy se v každém kroku zdvojnásobuje

SF nekombinující sítě:

- počet paketů v síti je výrazně větší než u kombinujícího modelu (každý paket je doručován induviduálně)
- plně duplexní kanály odpovídají orientovaným hranám tam i zpět
-
- 1-portový model: hamiltonovská kružnice
- všeportový plně duplexní model:
  - spodní mez $\rho = \lceil \frac{N-1}{d} \rceil$, kde *d* je minimální stupeň uzlu v G
  - 2D mřížka: každý uzel vyšle svůj paket oběma směry po hamiltonovské kružnici (všechny pakety se posunují podél jedné hamiltonovské kružnice)
  - toroidy
    - každý uzel rozdělí svůj paket na dvě poloviny
    - každý uzel vyšle svoje poloviny paketu oběma směry po dvou hamiltonovských kružnicích (tedy celkem 4 pakety - každou polovinu dvakrát)
    - když uzel obdrží obě dvojčata, složí je do původního paketu
    - časově hranově disjunktní kostry (každý uzel je kořenem nějaké kostry, která mu umožní odeslat paket všem uzlům) - lze použít i pro hyperkrychle

WH sítě:

1. rozděl G do 2^k souvislých regionů a zvol jejich reprezentanty
2. každý reprezentant shromáždí pakety uvnitř svého regionu použitím AOG
3. reprezentatni provedou AAB mezi sebou (simulací AAB v kombinující Q_k - sudo-liché a licho-sudé výměny)
4. každý reprezentant distribuuje globální info uvnitř svého regionu

AAS

Jedná se o úplnou výměnu - v síti je N · (N - 1) paketů.

SF nekombinující sítě: $\rho = \frac{\sum_{u \neq v} dist_G(u,v)}{2m}$, kde *m* je počet hran v síti

SF kombinující sítě:

- $\rho = diam(G)$
- 1-portová hyperkrychle: standardní výměna (půlení hyperkrychle, *n* kroků, v každém kroku párování v jedné dimenzi)
- 1D-toroid: cyklické obíhání

WH kombinující sítě:

- hyperkrychle: standardní výměna (půlení hyperkrychle, *n* kroků, v každém kroku párování v jedné dimenzi)
- mřížky a toroidy:
  - binární výměna: simulace binární výměny - v každém kroku mřížku půlíme (v obou směrech) a přenášíme informace z jedné půlky do druhé (je ale třeba více kroků kvůli zahlcení WH kanálů)
  - výměna mezi kvadranty: rekurzivně dělíme do kvadrantů a pokaždé provedeme mikro-AAS v obdélníku (osobně vyměníme informace mezi rohovými prvky, celkem tedy 3 kroky)

WH nekombinující sítě:

- hyperkrychle FD: přímá výměna (2^n - 1 permutací) - založena na přeložení (v j-tém kroku platí x = x xor j)
- mřížky a toroidy: uzlům přiřadíme adresy a simulujeme přímou výměnu