Systém s distribuovanou pamětí

- Obecně se skládá z uzlů a komunikačních kanálů
  - Uzel je prvek, na kterém probíhá nějaká část výpočtu
    - Typicky počítač v síti
  - Komunikační kanál je prvek, který přenáší data mezi sousedními uzly
    - Např. switch operující na síťové úrovni (KIV/UPS)

- Tři základní možnosti realizace
  - Univerzální počítačová síť
    - Software umožní počítače využít jako uzly
    - Ostatní funkce sítě jsou zachovány
  - Univerzální paralelní počítač
    - Hlavní úlohou je výpočet paralelní aplikace
    - Může počítat různé aplikace
    - Např. cluster
  - Jednoúčelový paralelní počítač
    - Je postavený na výpočet jedné konkrétní aplikace s maximálním možným urychlením
    - TWINKLE
      - The Weizmann Institute Key Locating Engine
      - Nekonveční elektro-optické analogové sumátory
      - Problém faktorizace velkých čísel u prolomení šifer se stává výpočetně přijatelnějším

- Obecně systém s distribuovanou pamětí umožňuje větší urychlení než systém se sdílenou pamětí díky paralelizaci komunikace
    - Zatímco se data přenášejí kanálem, uzel může počítat
    - Urychlení ovšem závisí na dalších parametrech
        - Objemu interakce
        - Celkovém objemu zpracovávaných dat
        - Konkrétní hw architektura
        - Jak dalece je použitý programový kód optimální pro danou architekturu

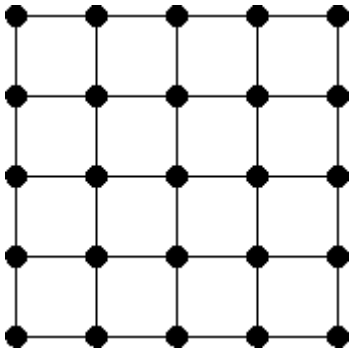
HW a síťový pohled na architekturu

- Topologie obecně
    - Pravidelná
        - síť má strukturu popsatelnou grafem pravidelné struktury
        - kruh, mřížka, krychle, …
    - Nepravidelná – např. Internet

- Směrování
    - Pevná topologie – můžete poslat vzkaz pouze sousedovi
    - Podle příjemce – směrování jak ho známe např. z IP
    - Podle odesílajícího – odesílající si určí cestu
        - IP „strict source and record route“ (SSRR)
        - IP „loose source and record route“ (LSRR)
            - Často blokováno z bezpečnostních důvodů – address spoofing (podvrhnutí adresy)

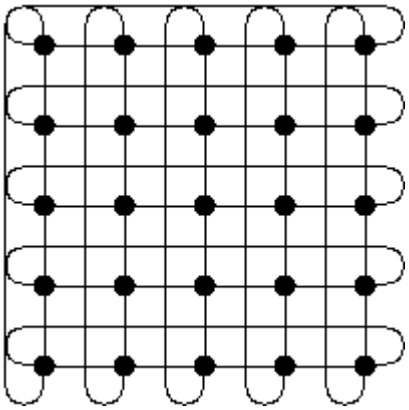
- Fyzická topologie
  - Pevná – procesory jsou spojeny komunikačním kanálem
  - Flexibilní
    - Circuit switching
    - Packet switching
      - Sériová linka s CSMA (KIV/UPS)

- Pevná topologie
  - každý s každým
  - 2D mřížka



http://www.cs.nmsu.edu/~pfeiffer/classes/573/notes/topology.html

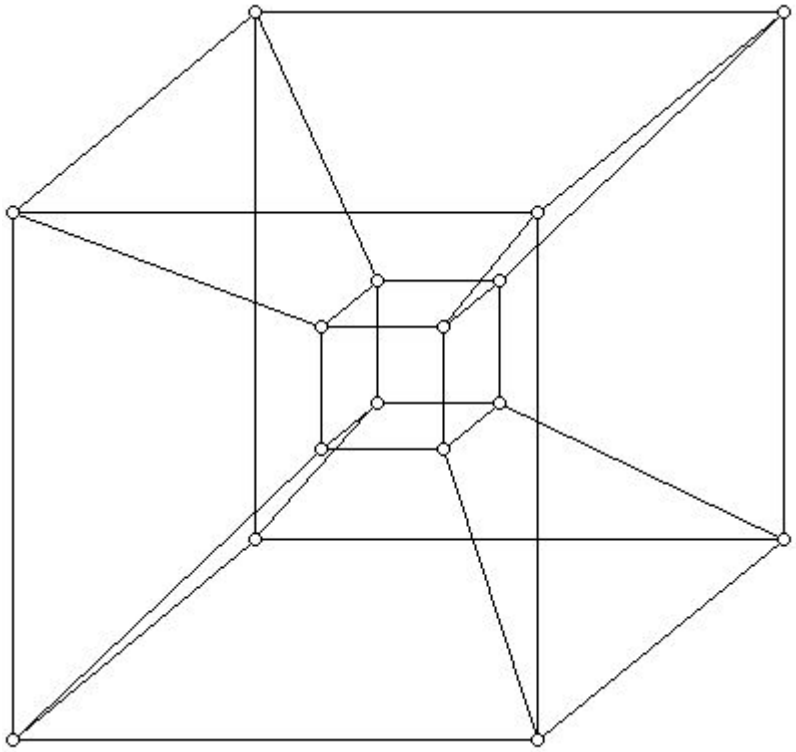
  - Toroid



http://www.cs.nmsu.edu/~pfeiffer/classes/573/notes/topology.html

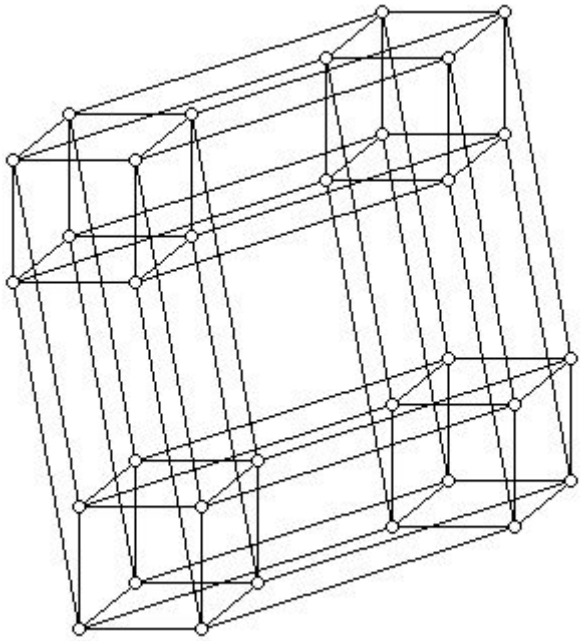
  - 3D mřížka
    - odpovídá fyzikálnímu 3D prostoru
    - krychle

- n-rozměrná krychle – n-Cube
  - 4-rozměrná krychle



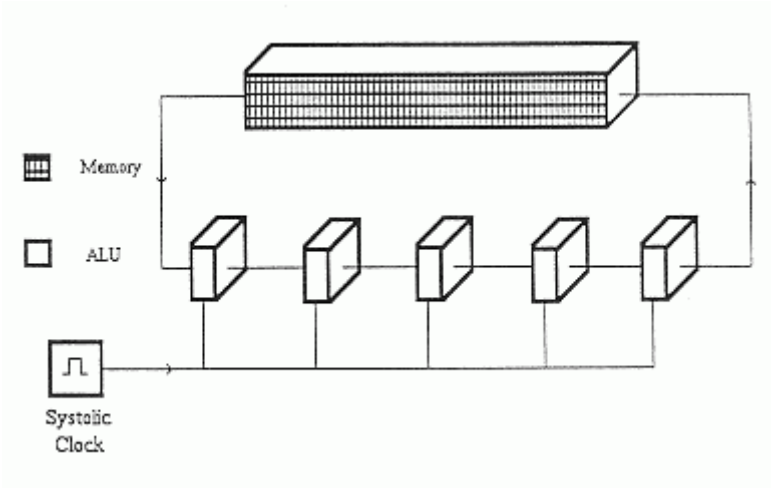
http://york.cuny.edu/~malk/tidbits/n-cube-tidbit.html

  - 5-rozměrná krychle

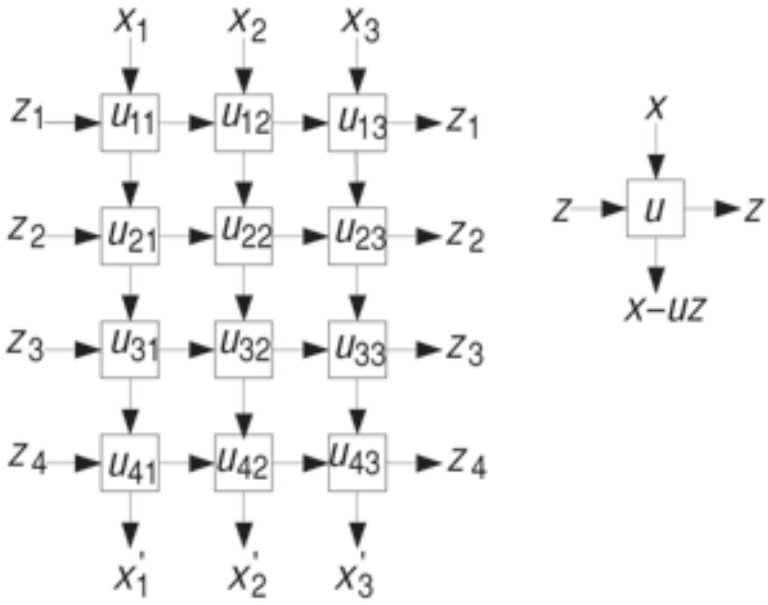


http://york.cuny.edu/~malk/tidbits/n-cube-tidbit.html

- Systolické pole
  - Několik vzájemně propojených uzlů, které si postupně s každým taktem hodin předávají data k dalšímu výpočtu
    - Synchronní, lock-stepped
    - Protějšek von Neumannovy architektury
      - Výpočet řídí data, ne program counter
  - Komplexnější pipeline
  - Komunikační kanály používají simplex



http://www.gigaflop.demon.co.uk/comp/chapt2.htm



http://www.cotsjournalonline.com/archive_images/cots0501_ss_4_1.gif

- Parametry
  - N
    - Celkový počet uzlů v síti
  - d_{ij}
    - vzdálenost mezi dvěma uzly
    - sousedé mají 1
  - d_{max}
    - nejhorší varianta, kolika uzly musí projít zpráva, než je doručena
    - nejdelší cesta zprávy v celém systému
  - počet sousedů
    - s kolika dalšími uzly je daný uzel spojen přímo
  - přenosová kapacita
    - agregovaně – kolik uzlů může najednou posílat zprávu
    - odolnost proti chybám – kolik komunikačních kanálů musí selhat, než se z jedné sítě stanou dvě
  - Snahou je dosáhnout
    - Co největšího počtu uzlů v síti
      - Škálovatelnost
    - Co nejmenší komunikační vzdálenosti – d_{max}
      - Tj. omezit komunikační zpoždění
    - Co nejmenší počet sousedů
      - Aneb, i komunikační kanál něco stojí
    - Dosáhnout co největší přenosové rychlosti

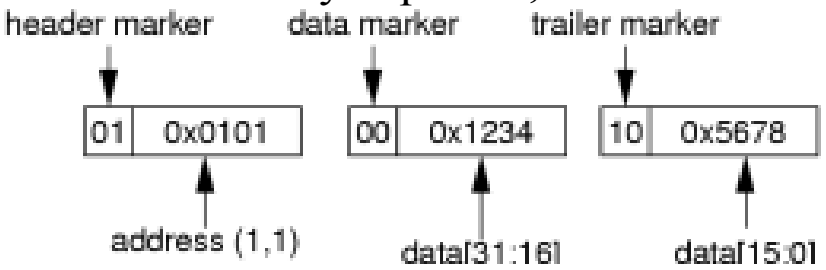
Fyzická adresa uzlu

- Sekvence bytů s nějakým významem
  - Např. 2x integer pro mřížku
    - `struct addr {int x, y;}`
- měla by souviset s polohou uzlu v síti
- mělo by z ní být možné odvodit fyzické adresy sousedů

- komunikační operace
  - pokud nelze docílit, aby spolu komunikovali pouze sousedé, pak

  - Store & Forward – přijme se celý paket, aby se poté poslal dál

  - Wormhole Switching/aka Routing
    - paket je rozdělen na menší jednotky
      - flit – flow unit
    - první flit obsahuje adresu, podle které se určí rozhraní odchozího komunikačního kanálu, kam jsou automaticky poslány zbývající flits daného paketu
    - zmenšuje komunikační zpoždění, ale vytváří nové možnosti k uvíznutí
      - velikost bufferů rozhraní vs. rychlost zpracování vs. počet simultánních flitů z různých paketů, atd.



http://www.es.ele.tue.nl/mininoc/doc/module_network2x2.htm

- pokud sestavujeme vlastní paralelní počítač, např. jednoúčelový, je to věc našeho návrhu
- ale co když máme využít např. IP?
  - Tj. univerzální počítačová síť a adresa, ze které lze odvodit sousední uzly, jak je má vidět výpočet

- Na vlastním segmentu bychom mohli zadat vlastní MAC, ale k čemu?
- Vlastní adresní rozsah, kdy číslo počítače budou zakódované souřadnice uzlu
  - Pak by také ale měly odpovídat umístění počítače v síti
- Prezentovaná síťová topologie
  - Fyzická topologie, tj. jak je nataženo vedení, je odlišná od síťové topologie, kterou vidí výpočet
  - Existuje bijektivní funkce, která mapuje mezi fyzickou a síťovou topologií
    - Může to být funkce, která se podívá do tabulky;-)

SW pohled na architekturu

- Alokování uzlů
  - 1 proces na 1 uzel
    - Např. pevně daná u paralelního počítače
    - 1 proces dokáže plně využít celý uzel, takže nemá smysl jich na jednom uzlu spouštět několik
    - Když OS uzlu neumí spustit více jak jeden proces najednou
  - Potenciálně nula až několik procesů na jeden uzel
    - Podrobněji bude na pozdější přednášce
  - Přidělení celé sítě pro jeden výpočet
    - Celkový čas výpočtu je pak dán
      - Dobou k zavedení programů, spuštění procesů a distribuce dat do uzlů
      - Vlastním výpočetem
      - Získáním výsledků z uzlů
  - Pro jeden výpočet bude přidělena pouze část sítě
  - Síť bude přidělena pro několik paralelně běžících výpočtů
    - Na jednom uzlu může běžet několik procesů
    - Nelze se spoléhat na odvozená urychlení, protože ta nepočítala se zátěží, kterou vygeneruje neznámý kód
    - Nehodí se pro synchronní/lock-stepped algoritmy – na společném uzlu by dva spolupracující procesy na sebe musely čekat dobu výpočtu jednoho kroku
    - Někdy to však může být výhodné – viz později

- Identifikace procesů
  - Každý z nich má jedinečné ID
  - ID může mít další aplikačně-orientovaný význam

  - Základní operace pro interakci jsou
    - send
    - receive
    - viz 1. přednáška

  - Cokoliv dalšího využívá právě tyto dvě operace

  - Pokud je proces vázán na konkrétní uzel a je tam sám
    - ID procesu odpovídá přímo adrese uzlu
    - Není problém s adresováním

  - Pokud je proces vázán na konkrétní uzel, ale na uzlu může současně běžet několik procesů
    - ID procesu odpovídá přímo adrese uzlu, ale navíc musí být ještě schopno přesně určit konkrétní proces
    - Stále ještě není problém s adresováním

  - Proces není vázán na konkrétní uzel
    - Aneb, proces může během výpočtu odmigrovat na jiný uzel
    - Pouze z ID procesu se nedá určit, na kterém uzlu se nachází
    - Distribuovaná aplikace si musí vést záznamy o tom, kde se který proces právě nachází a ty musí být konzistentní

        - Tabulka umístění procesů
            - buď jedna centrální,
            - nebo několik replik
                - rychlejší, ale náročnější

        - Vhodný middleware, který se o to „magicky“ postará
            - Pro programátora je příjemné adresovat data procesu a adresaci konkrétnímu uzlu přenechat jinému kódu, např. komunikační knihovně

        - Využít schopností programovatelné sítě
            - Např. Aktivní sítě – viz pozdější přednáška

- Virtuální topologie/komunikační schéma
    - Fyzická topologie – jak je to „sdrátováno“
    - Síťová topologie – jak vidí fyzickou topologii sw
    - Virtuální topologie – komunikační vazby procesů

    - Virtuální topologie může mít
        - Pravidelnou strukturu, např.
            - Mřížka
            - Hvězda – model farmer-worker
        - Nepravidelnou strukturu

    - Může být
        - Statická – např. mřížka
        - Dynamická
            - Procesy mohou procházet různými fázemi během výpočtu
            - Procesy mohou dynam. vznikat i zanikat

- Orientovaný graf
  - Uzly jsou procesy
  - Orientace určuje příjemce
  - Ohodnocení hrany určuje objem komunikace

- Cílem je namapovat virtuální topologii na síťovou tak, aby docházelo k co nejmenšímu zpoždění
  - Ideálně se fyzická, síťová i virtuální topologie shodují 1:1
  - „Úplně ideálně“ jde zároveň i o nejoptimálnější metodu výpočtu (který používá danou topologii)

  - Např. mřížka na mřížku o stejných, nebo větších, rozměrech
    - Úlohy tzv. geometrické dekompozice v rovině

  - Model Farmer-worker/Master-slave
    - Farmer úkoluje workery
    - Virtuální topologie je typu hvězda
    - Jenomže hvězdu lze takto namapovat jenom do určitého počtu procesů
      - U 2D mřížky 5 – 1 farmer a 4x soused worker
      - Co s tím?
        - Doručení na některé uzly projde přes několik uzlů
        - Jestliže se doručení zprávy podepíše na celkovém výkonu uzlu, pak v závislosti na režii zvážit rezervaci některých uzlů jenom pro komunikaci

Různé topologie

N = 1024	d_{max}	Poč. sousedů	Rozměr sítě
2D mřížka m*m	2x31 = 62	4	m = 32
n-krychle	10	10	n = 10
Každý s každým	1	1023	N

- 2D mřížka
  - Čtverec o hraně 32 uzlů – 32*32=1024
  - 4 sousedé, pokud není na okraji
    - 4 sousedé vždy, pokud jde o toroid
  - 2x31
    - Nejedná se o toroid
      - 2x16 pro toroid
    - Nejdéle trvá doručit zprávu do protilehlého vrcholu po diagonále
      - 31 po x a 31 po y

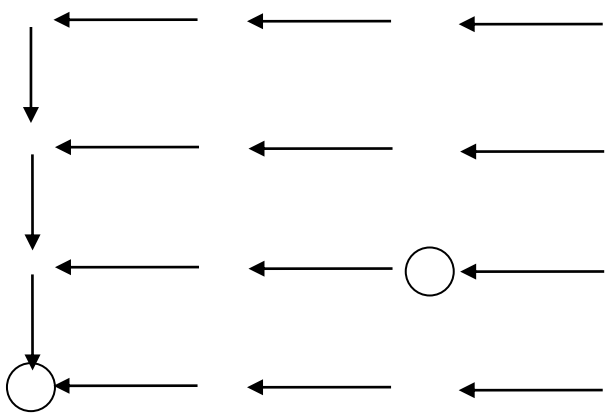
- n-krychle
  - N = 2^n
  - Pro n>3 je to náročné na představivost
    - 2D mřížka je n = 2
    - Pro n=3 si je to ještě možné představit a vidět, že vše vyjde 3

  - 10 sousedů
    - Souřadnice vrcholu je [n1, n2, …, n10]
    - Hranu tvoří 2 vrcholy, tedy n nabývají hodnot 0, nebo 1
      - A přitom 1 má pro každou souřadnici pouze jedno n ze všech
      - Tedy, 10 sousedů

- 10 jako největší vzdálenost
  - Začneme ve vrcholu [0, 0, 0, …, 0] a potřebujeme doručit zprávu do vrcholu [1, 1, 1, …, 1]
  - Jedna hrana znamená změnu pouze jednoho *n* z 0 na 1
  - Celkově tak musíme použít n hran, abychom změnili všechny 0 na 1
    - A těch bude právě 10 v tomto případě

Součet čísel v 2D mřížce

- SPMD

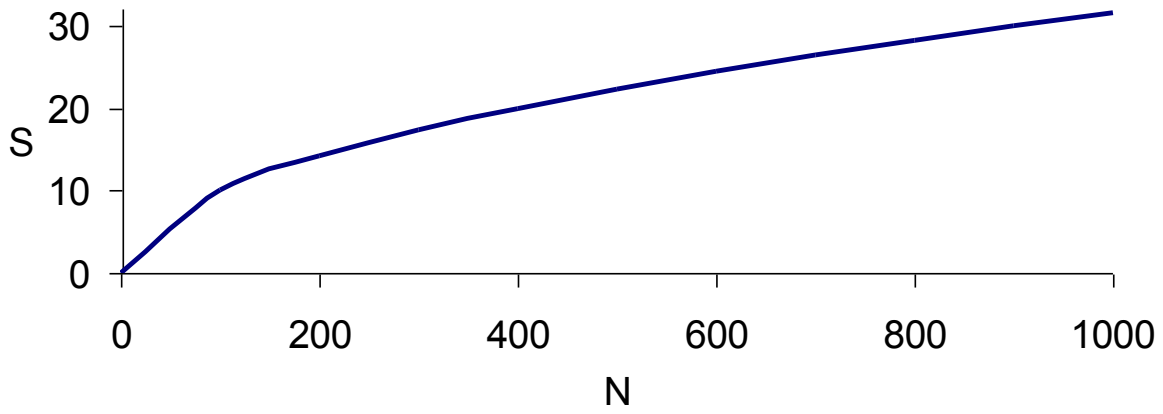


- Cílový uzel s výsledkem je v rohovém uzlu
  - Řádky se sečtou ve třech krocích a jejich součty se dostanou do krajních uzlů
  - Mezisoučty z krajních uzlů se sečtou v dalších třech krocích
  - Celkem je potřeba 6 kroků
    - O (2x(m-1)) => O(m) => O (N^0.5)

- Cílový uzel je co nejblíže středu
  - m/2 operací na součet řádku/sloupce

- urychlení
  - sekvenčně $t_{seq} = (m^2 - 1) \times t_{sum}$
  - paralelně $t_{par} = \left(\frac{m}{2} + m - 1 \right) \times t_{sum}$
  - t_{sum} je čas operace součtu
    - nebo jiné, náročnější operace, kterou bychom ve skutečnosti prováděli
  - čas přenosu zanedbáme
  - počet prvků pošleme k nekonečnu
    - tj. $m \to \infty$
  - urychlení

$$S = \frac{t_{seq}}{t_{par}} = \frac{m^2}{m} = m = \sqrt{N}$$



- Realističtější případ by mohla být transformace bitmapy
  - Každý uzel nejprve najde se své části lokální minimum a maximum jasu
  - Některý z uzlů určí globální minimum a maximum
    - Ty se pak rozešlou všem uzlům
  - Každý uzel provede transformaci své části bitmapy