PS Algorithmen für verteilte Systeme

eingereicht von

Baumgartner Dominik, Dafir Samy

Gruppe 1(16:00)

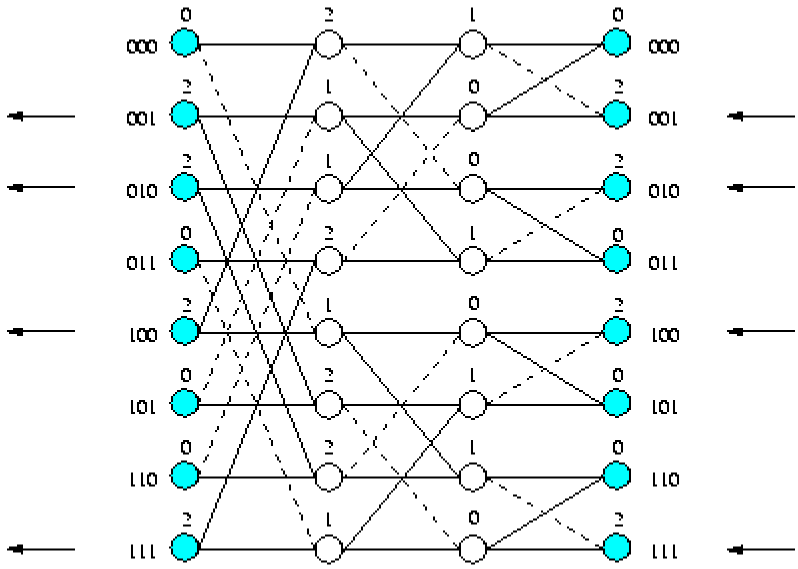
Aufgabe 6: Zeigen Sie, dass $CCC(k)$ Teilgraph des $BF(k)$ ist.
Z.z.: Für jedes n $\in \mathbb{N}$ gilt: $CCC(n)$ ist ein Teilgraph von $BF(n)$.

Bew.:
Sei Sei n $\in \mathbb{N}$. Definieren folgende Funktion:

$$p : V \to V \mid p((i, b)) := ((i + k(b)) mod(n), b).$$

wobei i das aktuelle Level ist und b ein Bitfolge der Länge n. Zudem gilt:
$k(b) := \begin{cases} 1, \text{ wenn b ungerade Anzahl an 1er Bits besitzt} \\ 0, \text{ sonst} \end{cases}$

Wird nun die Funktion v auf den $CCC(n)$ angewendet, werden alle Knoten von $CCC(n)$ auf $BF(n)$ abgebildet. Anhand der folgenden Abbildung ist zu erkennen, dass die Funktion v auch bijektiv ist.



Noch zu zeigen, dass auch jede Kante von $CCC(n)$ auf eine Kante von $BF(n)$ abgebildet wird:
$\forall e = \{u, v\} \in E_{CCC}$ gilt $p(e) := \{p(u), p(v)\} \in E_{BF}$.

Für jedes $e \in E_{CCC}$ gibt es zwei Fälle:
1. Fall:
$x = \{(i, b), ((i+1) mod(n), b)\} \in E_C$. Falls $k(w) = 0$ ist, so ist $p(x) = x$ und anderenfalls ist $p(x) = \{((i+1)\ mod(\text{n}), b), ((i+2)\ mod(n), b)\}$. In beiden Fällen gehört $p(x)$ zu E_C.
2. Fall:
$x = \{(i, b), (i, b(i))\} \in E_H$, wobei $b(i)$ die Bitfolge b mit geflippten Bit an Stelle i ist. Falls $k(w) = 0$ gilt, so ist $k(w(i)) = 1$ und
$p(x) = \{(i, b), ((i+1)\ mod(n), b(i))\}$.
Sonst ist $p(x)$=$\{((i+1)\ mod(n), b), (i, b(i))\}$. In beiden Fällen ist $p(x) \in E_X$.

Aufgabe 7: Zeigen Sie, dass das Butterfly-Netzwerk $BF(k)$ knotensymmetrisch ist.

7) 2.2. BF(k) ist knotensymmetrisch

Beweis:

Verwende Shuffle-Funktion

$s(u)$: zyklischer Links-shift von u. $s: \{0,1\}^k \to \{0,1\}^k$

$\varphi: V_k \to V_k \quad \varphi((i,u)) = ((i+1) \bmod k,\ s(u))$

- 2.2. φ ist ein Automorphismus
  - 2.2. φ ist bijektiv
    - 2.2. φ ist injektiv

$(i,u) \neq (j,v) \Rightarrow ((i+1) \bmod k,\ s(u)) \neq ((j+1) \bmod k,\ s(v))$

falls $i \neq j \Rightarrow (i+1) \bmod k \neq (j+1) \bmod k$

falls $i = j$: $u \neq v \Rightarrow s(u) \neq s(v)$

$\Rightarrow \varphi$ injektiv $|V_k| = |V_k| \Rightarrow \varphi$ surjektiv $\Rightarrow \varphi$ bijektiv

- 2.2. Kanten werden auf Kanten abgebildet (Kreis- & BF-Kanten)
  - Kreiskanten:

Sei $((i,u), ((i+1) \bmod k, u))$ eine Kreiskante

~~((i+1) mo~~

$\Rightarrow (((i+1) \bmod k, s(u)), ((i+2) \bmod k, s(u)))$ Ist auch eine Kreiskante

(beide haben gleiche geshiftete bits + Levels unterscheiden sich um 1)

  - ~~Hypercubeko~~ Butterflykanten:

Sei $((i,u), ((i+1) \bmod k, u(i)))$ eine BF-Kante

Dann ist auch $(((i+1) \bmod k,\ s(u)), ((i+2) \bmod k,\ s(u(i))))$ eine BF-Kante. Begründung:

- Die Levels der Knoten unterscheiden sich um 1
- u und $u(i)$ unterscheiden sich ~~um~~ in einer Bitstelle.

  $\Rightarrow$ auch $s(u)$ und $s(u(i))$ unterscheiden sich in einer Stelle (wurden nur geshiftet)

$\Rightarrow$ BF-Kanten werden auch korrekt abgebildet

- 2.2. Nicht vorhandene Kanten werden nicht auf Kanten abgebildet.

Kanten müssen folgende Eigenschaften erfüllen:
Kreiskanten: gleiche Bitkombination, Levels d. Knoten um 1 verschieden

BF-Kanten: Bitkombinationen unterscheiden sich in einem bit + Levels d. Knoten um 1 verschieden

- kreiskanten: $((i,u),(\overset{j}{\cancel{(i+1)}} \cancel{\bmod k}, \overset{v}{\cancel{u(k)}}))$

$u \neq v \Rightarrow$ Keine Kreiskante $\Rightarrow s(u) \neq s(v) \Rightarrow$ keine kreiskante

j & i nicht um 1 verschieden $(j+1) \bmod k$ & $(i+1) \bmod k$ auch nicht

$\Rightarrow$ keine Kreiskante. (weder vor noch nach Abbildung

- BF-Kanten $((i,u),(\overset{j}{\cancel{(i+1)}} \cancel{\bmod k}, v))$

Falls i & j sich nicht um 1 unterscheiden, unterschciden

sich auch $(i+1) \bmod k$ & $(j+1) \bmod k$ nicht um 1 Level

$\Rightarrow$ Vor & nach Abbildung keine kante mgl.

Falls u & v sich nich in 1 Stelle unterscheiden (in der i-ten

Stelle $v \neq u(i)$) unterscheiden sich auch $s(u)$ und $s(v)$ ~~s(u(i))~~

nicht in der i-ten Stelle (nach Abbildung $i+1$ te Stelle),

also $s(u)$ & $s(v)$ entweder nicht um 1 Stelle unterschiedlich oder

nicht in i-ter Stelle

$\Rightarrow$ Vor und nach Abbildung keine BF-Kante

φ ist ein Automorphismus

- $\delta_j : V_k \to V_k$; $\delta_j(i,u) = (i, u(j))$ ist bijektiv (Laut VO)

kreiskanten:

$(\delta_j(i,u), \delta_j((i+1) \bmod k, u)) \Rightarrow ((i,u(j)),((i+1) \bmod k, u(j)) \in E_k$
$\Rightarrow$ kreiskante

BF-Kanten:

$(\delta_j(i,u), \delta_j((i+1) \bmod k, u(i))) \Rightarrow ((i,u(j)),((i+1) \bmod k, u(i)(j))) \in E_k$

kanten unterscheiden sich um 1 Level & 1 bit da δ_j das gleiche bit flippt

$\Rightarrow \delta_j$ ist ein Automorphismus

- (i,u) auf (j,v) abbilden:

φ anwenden bis $j = i$. Position (j,x); $x \neq v$. Jetzt δ_j

auf alle bits in x anwenden die ungleich v sind

$\Rightarrow$ bei (j,v) angelangt.

Aufgabe 5: Schreiben Sie ein Programm, dass das folgende Viceroy-Netzwerk erstellt. Weisen Sie zunächst 100 Knoten 5 Ringen zu, wie in der Vorlesung beschrieben. Erstellen Sie anschließend die in der Vorlesung definierten Verbindungen zwischen diesen Knoten. Wählen Sie 100 Knotenpaare zufällig aus und berechnen Sie Routing-Pfade zwischen diesen Knotenpaaren. Bei der Berechnung eines Routing-Pfades dürfen Sie ausschließlich auf lokale Nachbarschaftsinformationen zurückgreifen. Geben Sie anschließend die Verteilung der Längen dieser Routing-Pfade aus.

In der folgenden Graphik ist die Verteilung der Längen der Routing-Pfade abgebildet.