Scheduling in wireless sensor networks WSNs

Petr Beránek
čtvrtek 14:30-16:00
Otevřená informatika - Počítačová grafika a interakce
beranpe6@fel.cvut.cz

I. Assignment

A. Problem Statement

Má semestrální práce se bude zabývat sítěmi bezdrátových senzorů (WSN).

Síť je tvořena bezdrátovými senzory. Lze ji reprezentovat jako orientovaný acyklický graf $G = (V, E)$, kde uzly V jsou jednotlivé senzory a E jsou hrany. Pořadí, ve kterém senzory budou vysílat, je omezeno hranami (E) v grafu G. Pokud hrana e vede z vrcholu v_i do vrcholu v_j, tak senzor v_i musí dokončit vysílání před tím, než v_j začne vysílat. Kanálů, na kterých senzory mohou vysílat, může být různé množství, a na každém kanálu může vysílat libovolné množství senzorů. Dalším omezením jsou také kolize. Pokud vysílají 2 senzory na stejném kanálu, mohou jejich vysílání kolidovat, což má za následek příjem špatných hodnot, tudíž kolize nesmí nastat. Pokud vysílají na různých kanálech, nemůže ke kolizím dojít. Které senzory kolidují se kterými je dáno maticí C o rozměrech $|V| * |V|$, kde prvky jsou $c_{ij} = 0$ pokud v_i a v_j nekolidují a $c_{ij} = 1$, pokud kolidují. Každý ze senzorů má také různé časy, jak dlouho musí vysílat. Tyto časy jsou předem známé, neměnné konstanty a každé vysílání musí celé doběhnout bez přerušování. Cílem mé práce je navrhnout přesný a heuristický algoritmus pro plánování těchto senzorových sítí, přičemž jde o to, aby perioda vysílání mohla být co nejkratší. (Aby doba od začátku vysílání prvního senzoru po odvysílání posledního senzoru byla co nejkratší.) Algoritmus bude založen na Multichannel Superframe Scheduling algoritmu popsaném v článku „Multichannel Superframe Scheduling for IEEE 802.15.4 Industrial Wireless Sensor Networks“[2].

B. Problem Categorization

Výsledný algoritmus má určit, kdy který senzor má vysílat a na kterém kanálu má vysílat. Pro každý senzor tedy potřebujeme získat dvě hodnoty. Počátek vysílání senzoru i s_i a kanál, na kterém bude vysílat ch_i. V této úloze jde o naplánování, kdy a jak má který senzor vysílat, je to tedy kategorie scheduling.

II. Related works

Tématem plánování sítí bezdrátových senzorů se zabývá například článek „Multichannel Superframe Scheduling for IEEE 802.15.4 Industrial Wireless Sensor Networks“[2], který jsem měl v plánu použít pro implementaci mého řešení. Naneštěstí algoritmus MSS nepočítá s precedencí senzorů (že senzor v_i musí dovysílat, než v_j začne, pokud graf $G = (V, E)$ obsahuje hranu $(v_i, v_j) \in E$). Dále také počítá s dostatečným množstvím kanálů, na kterých senzory mohou vysílat. Algoritmus se zabývá především kolizemi mezi senzory. MSS dále předpokládá, že celá síť bude tvořena zakořeněným stromem, přičemž PAN Coordiantor bude kořenem. To ale odporuje zadání mé práce[1], kde je očekáván pouze orientovaný acyklický graf. V zadání[1] je na ukázkovém příkladu přímo vidět, že graf není stromem a může mít více konců (vrcholy s výstupním stupněm rovným 0, vrcholy 2 a 5) a více zdrojů (vrcholů se vstupním stupněm rovným 0, vrcholy 1, 4, 6 a 7). Obdobným tématem se zabývá i článek[3]. Ten je především zaměřen na spolehlivost, ale jejich formulace ILP úlohy je poměrně zajímavá a trochu jsem se jí inspiroval.

Většina článků naneštěstí počítá jen s některým podmínkami ze zadání a počítá s jinými. Většinou počítají s tím, že síť bude mít pouze jeden sink[4][5], nebo nepočítají s předchůdci v síti[2].

III. Problem solution

A. Design

Vstupními parametry pro tuto úlohu jsou:
Skalár n rovný počtu senzorů.

Skalár m rovný počtu kanálů.

Vektor SD o velikosti n prvků. Prvek SD_i říká kolik senzor i vyžaduje časových kvant pro dokončení přenosu.

Matice P (matice předchůdců), kde prvek p_{ij} bude nabývat hodnoty 1 tehdy, pokud senzor v_i musí dovysílat dříve, než senzor v_j začne vysílat, jinak nabývá hodnoty 0.

Matice C (matice kolizí), kde prvek c_{ij} bude nabývat hodnoty 1 tehdy, pokud senzory v_i a v_j spolu kolidují, pokud vysílají na stejném kanálu, jinak nabývá hodnoty 0.

Rozhodl jsem se řešit tuto úlohu pomoí ILP modifikací algoritmu pro jednoduché plánování představeného na přednáškách. Do této formulace bylo nezbytné zahrnout všechny podmínky tohoto problému.
Základem je minimalizace času, který budou všechny senzory potřebovat od začátku vysílání prvního senzoru po skončení vysílání posledního senzoru. To lze formulovat jako:

$$min(z)$$

$$s_i + SD_i <= z \qquad i \in \{1..n\}$$

Proměnná s_i vyjadřuje začátek vysílání senzoru i.
Dále je nezbytné zachovat pořadí senzorů takové, že splní podmínky dané grafem G (zde definovýny pomocí matice P). To je možné formulovat nerovnicemi:

$$if\ p_{ij} = 1: \quad s_i + SD_i <= s_j \qquad i, j \in \{1..n\}$$

Každá z těchto nerovnic bude přidána pouze v případě, že prvek $p_{ij}) = 1$. Jelikož obsah matice P je znám ještě před začátkem algoritmu, je možné už předem určit, kolik rovnic bude použito.

Největším problémem této úlohy bylo sestavení nerovnic pro vyřešení kolizí mezi senzory, pokud vysílají na stejném kanálu. K tomu je potřeba přidat proměnnou ch_i, která bude určovat na kterém kanálu bude senzor i vysílat. Tato proměnná bude nabývat hodnot $\{1..m\}$. Potom lze zapsat podmínku jako:

$$if\ c_{ij} = 1\ \wedge\ ch_i = ch_j: \qquad s_i + SD_i \leq s_j \quad or \quad s_j + SD_j \leq s_i \qquad i, j \in \{1..n\}$$

Prvním problémem v této podmínce je fakt, že jsou tu dvě nerovnice a musí platit pouze jedna z nich. Proto jsem přidal rozhodovcí proměnnou q_{ij} a dostatečně velikou konstantu Q, aby byla jedna z nerovnic vždy vypnuta. Tím vznikl tvar:

$$\begin{aligned} &if\ c_{ij} = 1\ \wedge\ ch_i = ch_j: \\ &s_i + SD_i \leq s_j + Q * q_{ij} \qquad && i, j \in \{1..n\} \\ &s_j + SD_j \leq s_i + Q * (1 - q_{ij}) && i, j \in \{1..n\} \end{aligned}$$

Nyní bylo třeba vyřešit problém s podmínkou $ch_i = ch_j$. Jelikož podmínka obsahuje proměnné, nejsme schopni předem říci, zda tato podmínka v úloze být má, nebo ne. To je možné přeformulovat s použitím absolutní hodnoty $abs(x)$ a konstanty R dostatečně vysoké na to, aby byla rovnice vždy platná, pokud absolutní hodnota bude různá od nuly.

$$\begin{aligned} &if\ c_{ij} = 1: \\ &s_i + SD_i \leq s_j + Q * q_{ij} \qquad\quad + R * abs(ch_i - ch_j) \qquad && i, j \in \{1..n\} \\ &s_j + SD_j \leq s_i + Q * (1 - q_{ij}) + R * abs(ch_i - ch_j) && i, j \in \{1..n\} \end{aligned}$$

Tato úprava ale problém nevyřešila, neboť absolutní hodnotu není možné použít a musí se její použití nějak obejít. Řešení tohoto problému mi trvalo nejdéle a musel jsem k tomu využít dalších pomocných rozhodovacích proměnných. Absolutní hodnotu $abs(a-b)$ je možné rozepsat na dvě možnosti $a-b$ a $b-a$. Obdobným způsobem jsem zdvojil počet rovnic na:

$$
\begin{aligned}
s_i + SD_i &\le s_j + Q * q_{ij} &&+ R * (ch_i - ch_j) && i,j \in \{1..n\} \\
s_i + SD_i &\le s_j + Q * q_{ij} &&+ R * (ch_j - ch_i) && i,j \in \{1..n\} \\
s_j + SD_j &\le s_i + Q * (1 - q_{ij}) &&+ R * (ch_i - ch_j) && i,j \in \{1..n\} \\
s_j + SD_j &\le s_i + Q * (1 - q_{ij}) &&+ R * (ch_j - ch_i) && i,j \in \{1..n\}
\end{aligned}
$$

V tomto případě, když proměnné ch_i a ch_j si jsou rovny, jejich rozdíl bude nulový a nerovnice budou stejné (1. shodná s 2. a 3. shodná se 4.). V případě nenulového rozdílu bude jedna dvojice nerovnic vždy splněna(1. a 3. nebo 2. a 4.) a o druhé nic nevíme. Rozdíl může nabývat hodnot od $n-1$ až do $1-n$. Když tedy zvolíme třeba $Q = 10 * n * R$, bude aktivní člen Q převažovat a nestane se, že by se z neaktivních nerovnic staly aktivní. Problémem je ovšem to, že potřebujeme, aby všechny 4 nerovnice byly v tomto případě vypnuty (vždy splněny), ale vypnutou máme pouze tři (ty s kladným rozdílem ch_i a ch_j a ty, kde je Q člen aktivní) a jedna je nesplnitelná. To je ale možné upravit přidáním další rozhodovací proměnné r_{ij} a konstanty S takové, že jednoznačně převáží R, například $S = 5 * n * R$. Dá se předpokládat, že v případě nulového rozdílu, bude jedno, která z nerovnic bude aktivní, protože jsou obě shodné. V případě nenulového rozdílu bude muset být vypnuta nerovnice, která byla nesplnitelná, aby podmínka mohla být splněna. Touto úpravou získáme nerovnice:

$$
if\ c_{ij} = 1:
$$

$$
\begin{aligned}
s_i + SD_i &\le s_j + Q * q_{ij} &&+ R * (ch_i - ch_j) + S * r_{ij} && i,j \in \{1..n\} \\
s_i + SD_i &\le s_j + Q * q_{ij} &&+ R * (ch_j - ch_i) + S * (1 - r_{ij}) && i,j \in \{1..n\} \\
s_j + SD_j &\le s_i + Q * (1 - q_{ij}) &&+ R * (ch_i - ch_j) + S * r_{ij} && i,j \in \{1..n\} \\
s_j + SD_j &\le s_i + Q * (1 - q_{ij}) &&+ R * (ch_j - ch_i) + S * (1 - r_{ij}) && i,j \in \{1..n\}
\end{aligned}
$$

Podmínka $c_{ij} = 1$ nám už nevadí, neboť c_{ij} jsou konstanty známé už na začátku. Tím získáváme formulaci:

$$
min(z)
$$

$$
s_i + SD_i \le z \qquad i \in \{1..n\}
$$

$$
if\ p_{ij} = 1:
$$

$$
s_i + SD_i \le s_j \qquad i,j \in \{1..n\}
$$

$$
if\ c_{ij} = 1:
$$

$$
\begin{aligned}
s_i + SD_i &\le s_j + Q * q_{ij} &&+ R * (ch_i - ch_j) + S * r_{ij} && i,j \in \{1..n\} \\
s_i + SD_i &\le s_j + Q * q_{ij} &&+ R * (ch_j - ch_i) + S * (1 - r_{ij}) && i,j \in \{1..n\} \\
s_j + SD_j &\le s_i + Q * (1 - q_{ij}) &&+ R * (ch_i - ch_j) + S * r_{ij} && i,j \in \{1..n\} \\
s_j + SD_j &\le s_i + Q * (1 - q_{ij}) &&+ R * (ch_j - ch_i) + S * (1 - r_{ij}) && i,j \in \{1..n\}
\end{aligned}
$$

$$
\begin{gathered}
z \in Z_0^+ \\
s_i \in Z_0^+ \qquad i \in \{1..n\} \\
ch_i \in \{1..m\} \qquad i \in \{1..n\} \\
q_{ij} \in \{1, 0\} \qquad i,j \in \{1..n\} \\
r_{ij} \in \{1, 0\} \qquad i,j \in \{1..n\}
\end{gathered}
$$

Takto získáme $1 + 2 * n + 2 * n^2$ proměnných a v nejhorším případě $n + 5 * n^2$ nerovnic.

B. *Implementation*

- Implementation details of your algorithm. ILP problem formulation/description of heuristics/important non-trivial data structures etc.
- Range: 0.5 to 1 page
- Deadline: 04.05. 2016

Exaktní algoritmus pro plánování byl naimplementován v matlabu s použitím TORSCHE Toolboxu a pro algoritmus byla využita ILP formulace kterou je možné vidět výše v části Design.

Heuristický algoritmus byl implementován v javě a nevyžaduje žádný solver ani nic specifického (je to pouze obyčejná java). Prvním krokem heuristického algoritmu je vypočtení priorit senzorů (uzlů grafu). Priorita udává pořadí senzorů, v jakém by měli být plánovány. Priorita senzoru x_i se vypočítá jako:

$$p_i = SD_i + max(p_j)$$

Kde p_j jsou priority senzorů x_j, které jsou přímými potomky senzoru x_i (následnost a precedence se bere z grafu G sítě senzorů). Takto každý senzor získá prioritu tak vysokou, jako je největší "délka cesty"od daného senzoru po orientovaných hranách k nejvdálenějšímu sinku (senzoru bez potomků). Délkou cesty je zde myšlen součet SD_i všech senzorů x_i takových, že jsou na dané cestě. Takto budou největší prioritu mít ty senzoru, které jsou nejvíce vzdálené od konce plánu. Pokud má senzor prioritu $p_i = 15$, znamená to, že do konce plánu bude potřeba nejméně 15 časových jednotek kvůli tomu, že jsou senzoru na sobě závislé precedencí. To znamená, že délka l nejkratšího možného plánu bude vždy větší nebo rovna maximální hodnotě priority a neexistuje plán, který by byl kratší.

Algoritmus dále pracuje iterativním procházením jednotlivých časových úseků. Algoritmus v nejhorším případě projde $\Sigma(SD_i))i \in \{1..n\}$ úseků (Případ, kdy by byly všechny senozry umístěny na stejný kanál za sebou.) což je nejdelší možná délka plánu.

Z množiny senzorů (vrcholů) grafu G je vybrána podmnožina P taková, že tyto senzory nemají v grafu G žádné předchůdce a tato podmnožina je seřazena dle priority. Pro každý časový úsek algoritmus projde množinu P a pokusí se přiřadit senzory v daném okamžiku k některému z kanálů. To je možné pouze pokud dovysílali všichni předci daného senzoru a je možné umístit senzor na některý kanál, kde nebude kolidovat s jiným senzorem. Pokud se žádný senzor nepodařilo přiřadit, algoritmus se přesune na další časový úsek a postup se opakuje. Pokud se podaří přiřadit nějaký senzor S ke kanálu, tak senzor S je odebrán z grafu G spolu se všemi hranami vedoucimi z tohoto uzlu. Naplánovaný senzor x_i zároveň obsadí kanál, na který je naplánován na následujících SD_i časových jednotek.

Plánování končí ve chvíli, kdy v podmnožině P nezbyde žádný senzor. To nastane v tom případě, že všechny senozry z G již byly naplánovány a tudíž je práce hotová, nebo se v grafu vyskytuje cyklus a plánování dalších senzorů už není možné.

Heuristický algoritmus je urychlen tak, že množina P nemusí být v každé iteraci budována znova. Pokud se nezdařilo naplánovat žádný senzor, množina P zůstane stejná. Pokud byl nějaký senzor S naplánován, tento senzor se z P odebere a zkontrolují se všichni potomci S jestli by neměli být přidáni do P a pokud ano, tak se přidají.

Heuristický algoritmus pro svůj výpočet vyžaduje udržet v paměti v nejhorším případě $\Sigma(SD_i) * m * k$ hodnot, přičemž k je největší počet senzorů, které spolu nekolidují (horní hranice toho, kolik by v nejhorším případě mohlo být na jednom kanálu senzorů, počítá se lokálně pro každý senzor (pro každý řádek matice kolizí se spočítá, kolik je v daném řádku 0 (nekolizí) a vybere se největší množství, neuvažují se precedence ani kolize mezi ostatními senzory)). To může být poměrně veliké množství hodnot (především pro veliké SD_i). Proto je vhodné před započetím algoritmu vydělit všechny SD_i jejich největším společným dělitelem pro zmenšení paměťové náročnosti.

IV. Experiments

A. *Benchmark Settings*

K testování obou algoritmů byly použity 4 sady dat. Tyto sady bylu vytvořeny ručně a tak, aby jejich řešení nemělo triviální výsledek (Aby nebylo příliš mnoho kanálů/příliš málo kolizí, aby senzory nebyly vázány pouze precedencí. Dále aby síť senzorů měla více počátků a konců, aby v každém okamžiku mohlo vysílat více senzorů najednou a nebylo možné pouze jediné uspořádání senzorů.) Testy heuristického algoritmu byly provedeny pro porovnání i s prioritou jednotlivých senzorů, tak bez ní (senzory se netřídí) pro ověření, zda tato úprava způsobí lepší výsledky. Testování probíhalo na stroji s těmito parametry:

MS Windows 7 Pro 64-bit
Intel Core i7-2670QM 2.2GHz
8GB RAM
NVIDIA GeForce GT 540M

B. Results

Testování proběhlo na 4 sadách dat. První sada obsahuje síť se 7 senzory a dvěmi kanály. Druhá 10 senzorů a jeden kanál. Třetí 20 senzorů a 3 kanály a čtvrtá 24 senzorů a 2 kanály. (První sada dat je úloha, která byla ukázána v zadání semestrální práce.)

Exact

sada	čas	konec plánu
dataset1	420ms	13
dataset2	54ms	22
dataset3	1m 42s	44
dataset4	14m 21s	35

Heuristic

sada	čas	konec plánu
dataset1	2ms	13
dataset2	2ms	22
dataset3	2ms	44
dataset4	2ms	35

Heuristic + Priority

sada	čas	konec plánu
dataset1	1ms	13
dataset2	1ms	23
dataset3	1ms	44
dataset4	1ms	41

C. Discussion

Během testování jsem narazil na zásadní chybu v exaktním algoritmu. Pokud byla vytvořena úloha, která měla pouze jeden kanál, tak proměnné ch_i byly shora i zdola omezeny stejnou hodnotou (a tudíž mely být konstantní). S tím si ovšem funkce $ilinprog$ z TORSCHE Toolboxu nedokázala poradit. Problém byl vyřešen redukcí problému. Jelikož je v úloze pouze jeden kanál, tak tuto část ILP formulace

$$\begin{aligned}
&if\ c_{ij} = 1 : \\
&s_i + SD_i \leq s_j + Q * q_{ij} \quad + R * (ch_i - ch_j) + S * r_{ij} && i, j \in \{1..n\} \\
&s_i + SD_i \leq s_j + Q * q_{ij} \quad + R * (ch_j - ch_i) + S * (1 - r_{ij}) && i, j \in \{1..n\} \\
&s_j + SD_j \leq s_i + Q * (1 - q_{ij}) + R * (ch_i - ch_j) + S * r_{ij} && i, j \in \{1..n\} \\
&s_j + SD_j \leq s_i + Q * (1 - q_{ij}) + R * (ch_j - ch_i) + S * (1 - r_{ij}) && i, j \in \{1..n\}
\end{aligned}$$

lze zjednodušit na:

$$\begin{aligned}
&if\ c_{ij} = 1 : \\
&s_i + SD_i \leq s_j + Q * q_{ij} && i, j \in \{1..n\} \\
&s_j + SD_j \leq s_i + Q * (1 - q_{ij}) && i, j \in \{1..n\}
\end{aligned}$$

neboť odpadá část formulující podmínku "pokud budou oba senzory vysílat na stejném kanálu"(Byla vynechána 2. a 4. rovnice a od 1. a 3. byly odebrány R a S členy.). Tím se dokonce zredukoval počet proměnných z $1+2*n+2*n^2$ na $1+n+n^2$ neboť není potřeba n proměnných ch_i označujících kanál senzoru a n^2 proměnných r_{ij} použitých pro vybrání správného znaménka rozdílu $|ch_i - ch_j|$.

Mé původní očekávání bylo, že exaktní algoritmus bude vytvářet nejoptimálnější plán za cenu, že bude trochu pomalejší. U heuristického algoritmu že bude velmi rychlý, ale že nebude dávat nejoptimálnější plán. U heuristického algoritmu bez třídění jsem očekával, že oproti heuristickému s tříděním bude dávat výrazně méně dobré výsledky.

Růst délky výpočtu exaktního algoritmu s množstvím senzorů mě opravdu překvapil. Jelikož časová náročnost rostě velmi rychle s přibývajícími proměnnými, pro více než 30 senzorů už prakticky nebude možné ho použít. Na druhou stranu mě heuristický algoritmus docela překvapil, že jeho výsledky byly nejen rychlé, ale také optimální stejně jako u exaktního algoritmu. Očekával jsem, že se také budou odchylovat od optima. To ale může být způsobeno vlastnostmi testovaných dat. Na druhou stranu odstranění třídění (a tudíž ignorace priorit senzorů) způsobilo rozdíl jen v jediném plánu. Očekával jsem, že se budou lišit i ostatn plány (především ten s 20 senzory).

V. CONCLUSION

Výsledkem této semestrální práce jsou dva algoritmy pro plánování sítí bezdrátových senzorů. Jeden umožňuje přesné řešení problému, ale za cenu poměrně dlouhé doby výpočtu, pro vyšší počet senzorů (přes 20). Druhý algoritmus umožňuje plánováni i větších množství senzorů ve velmi krátkém čase, ale výsledek nemusí být optimální. Přesný algoritmus ke svému běhu vyžaduje TORSCHE Toolbox pro matlab, heuristický algoritmus je napsán v čisté Javě bez jakékoliv potřeby dalších nástrojů/knihoven.

REFERENCE

[1] —, *“mix of problems“*[vid. 27.3.2016] Dostupné z: https://moodle.fel.cvut.cz/mod/resource/view.php?id=30511

[2] Toscano, Emanuele, and Lucia Lo Bello. *”Multichannel superframe scheduling for IEEE 802.15. 4 industrial wireless sensor networks.”* Industrial Informatics, IEEE Transactions on 8.2 (2012): 337-350.

[3] ZHANG, Xiaoling, et al. *”Reliable transmission scheduling for multi-channel wireless sensor networks with low-cost channel estimation.”* Communications, IET, 2013, 7.1: 71-81.

[4] ERGEN, Sinem Coleri; VARAIYA, Pravin. *”TDMA scheduling algorithms for wireless sensor networks.”* Wireless Networks, 2010, 16.4: 985-997.

[5] ZHANG, Xiaoling, et al. *”Joint routing, scheduling, and power control for multichannel wireless sensor networks with physical interference.”* Journal of Control Theory and Applications, 2011, 9.1: 93-105.