Skedulointi, kuormituksen tasaus, robotin navigaatio

Esitelmä algoritmiikan tutkimusseminaarissa

17.2.2003

Kimmo Palin

Tietojenkäsittelytieteen laitos

Helsingin Yliopisto

Sisältö

Tällä kertaa tarjolla toisiinsa liittymättömiä käytännön ongelmia otsikolla ’nice to know’.

- Töiden skedulointi.
- Rasituksen tasaus.
- Robotin navigointi tuntemattomassa ympäristössä.

Merkintöjä

- Jonon $\sigma = J_1, J_2, \ldots, J_n$ työt saapuvat online.
- Työllä J_k on aikatarve p_k.
- Saapuva työ skeduloidaan välittömästi jollekin m koneesta.

Perusongelmasta saadaan eri muunnoksia rippuen siitä,

- tunnetaanko p_k etukäteen vai ei,
- ovatko koneet identtisiä vai ei,
- ovatko kaikki koneet aina käytettävissä vai ei,
- mitä kohdefunktiota optimoidaan,
- ...

Skedulointi

Klassisessa tapauksessa prosessin aikatarve tunnetaan ja koneet ovat identtisiä. Tavoitteena on minimoida viimeisen työn valmistumiseen kuluva aika.

AHNE algoritmi: skeduloi saapuva työ vähiten kuormitetulle koneelle.

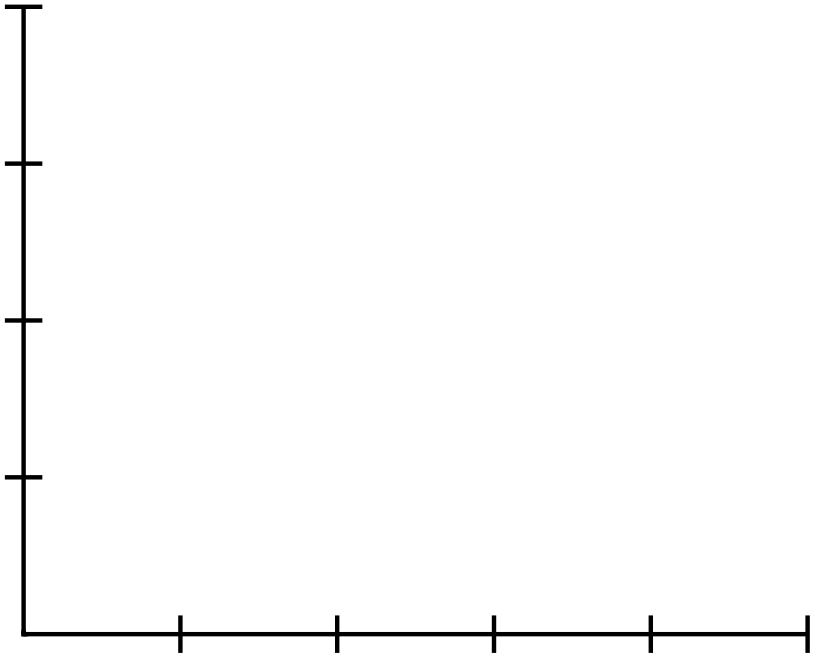
Skedulointi

Klassisessa tapauksessa prosessin aikatarve tunnetaan ja koneet ovat identtisiä. Tavoitteena on minimoida viimeisen työn valmistumiseen kuluva aika.

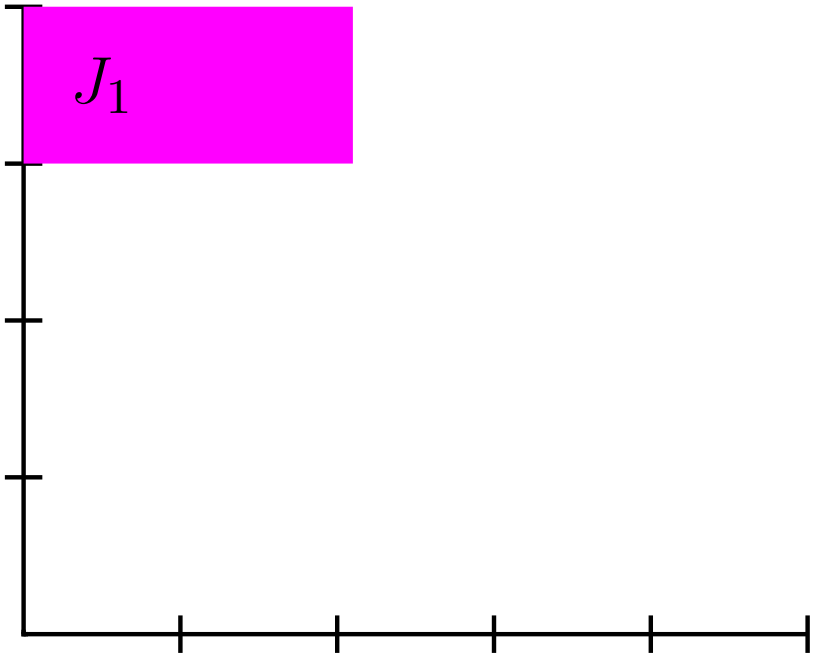
AHNE algoritmi: skeduloi saapuva työ vähiten kuormitetulle koneelle.

Lause 28 *AHNE algoritmi on* $(2 - \frac{1}{m})$ *kilpailukykyinen.*

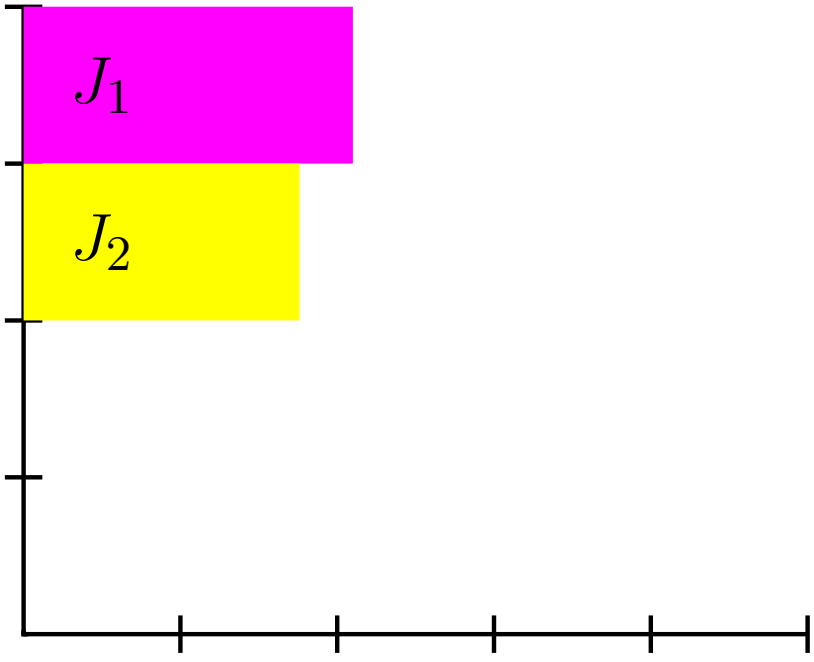
Todistus



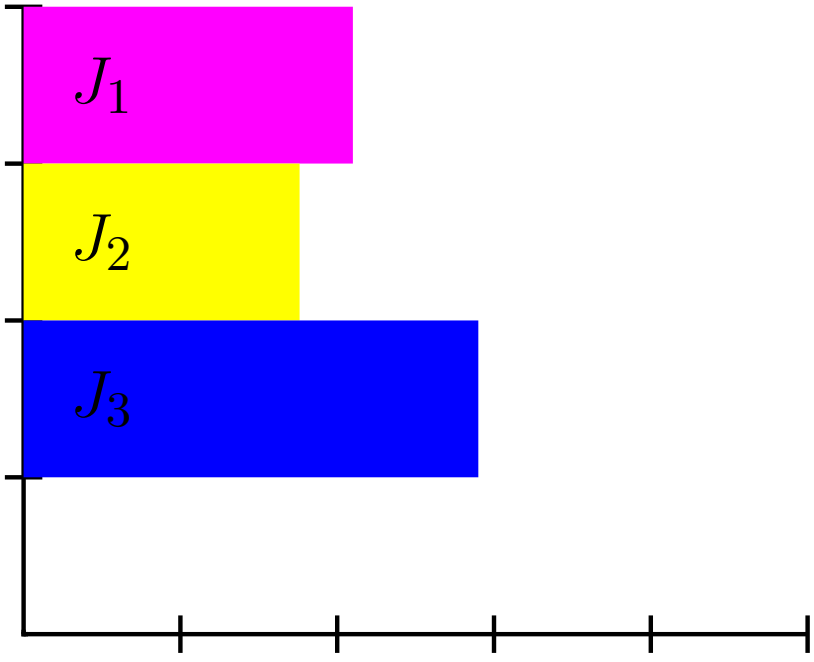
Todistus



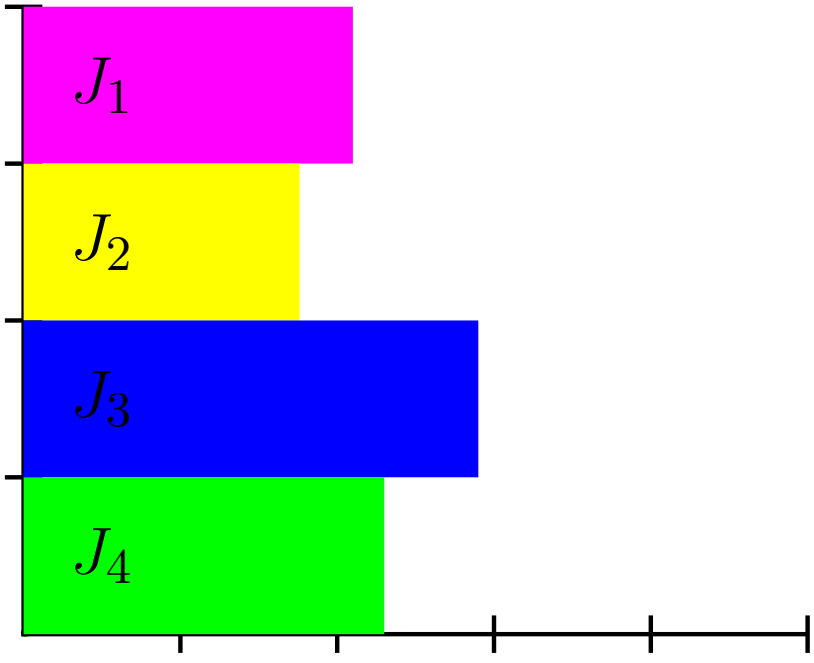
Todistus



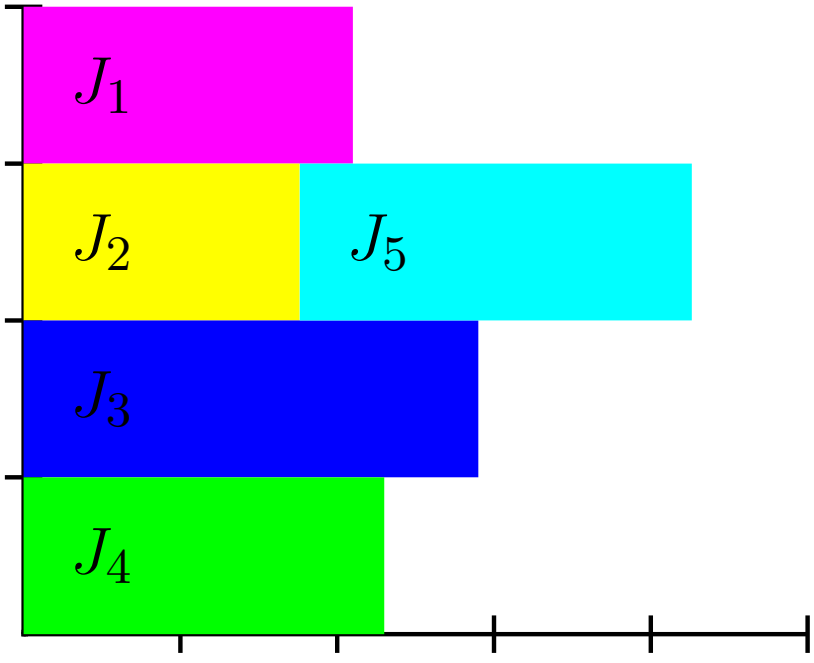
Todistus



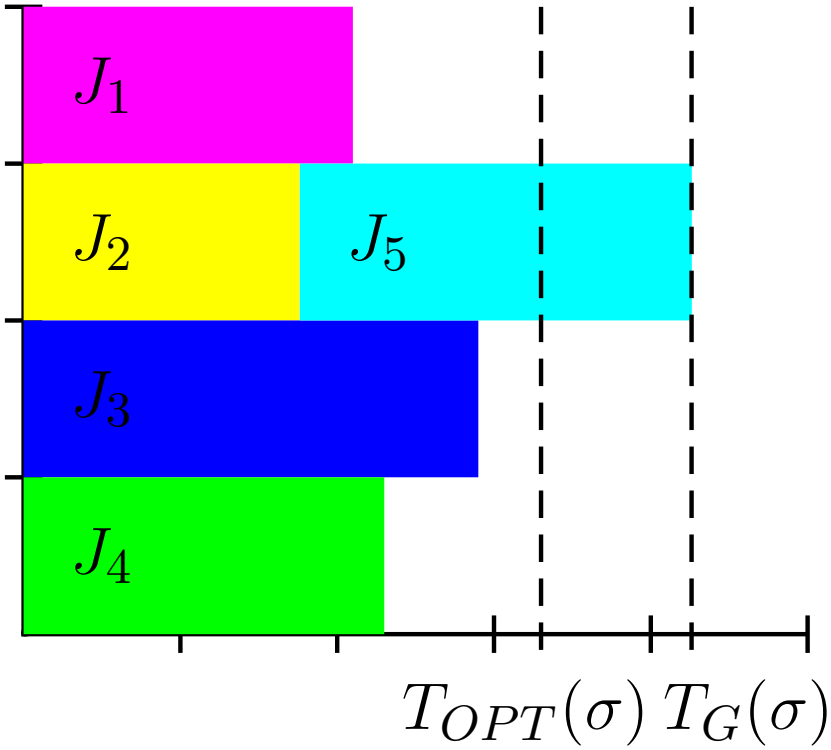
Todistus



Todistus



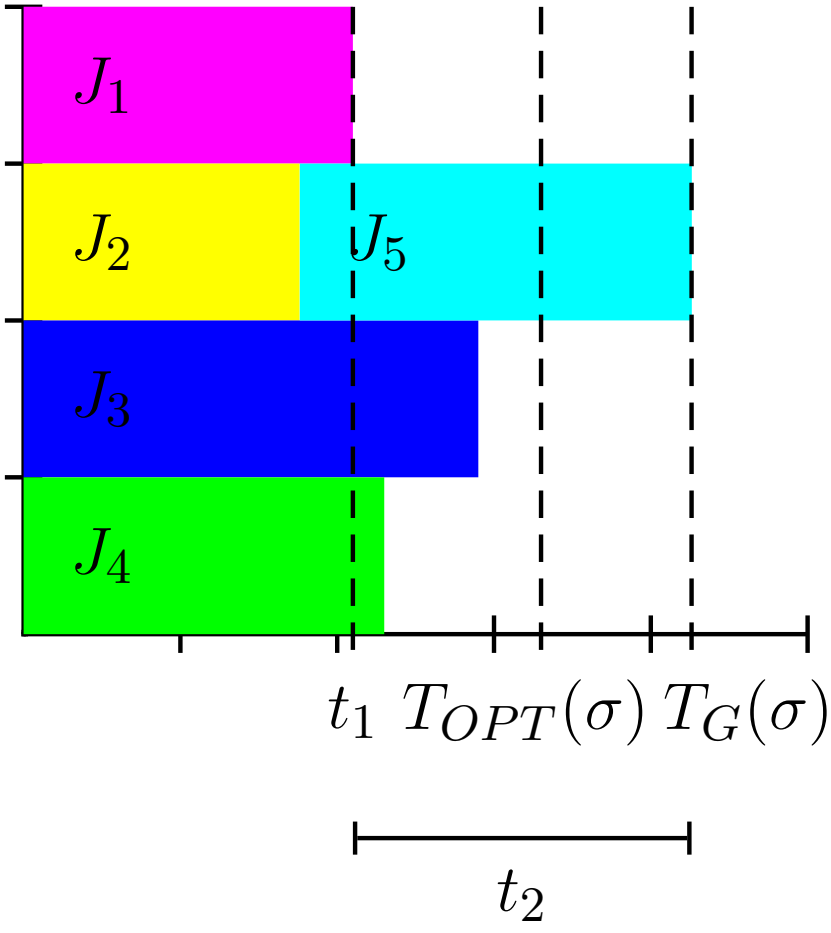
Todistus



$T_G(\sigma)$ työjonon valmistumisaika AHNE algoritmin skeduloimana.

$T_{OPT}(\sigma)$ optimaalinen valmistumisaika.

Todistus

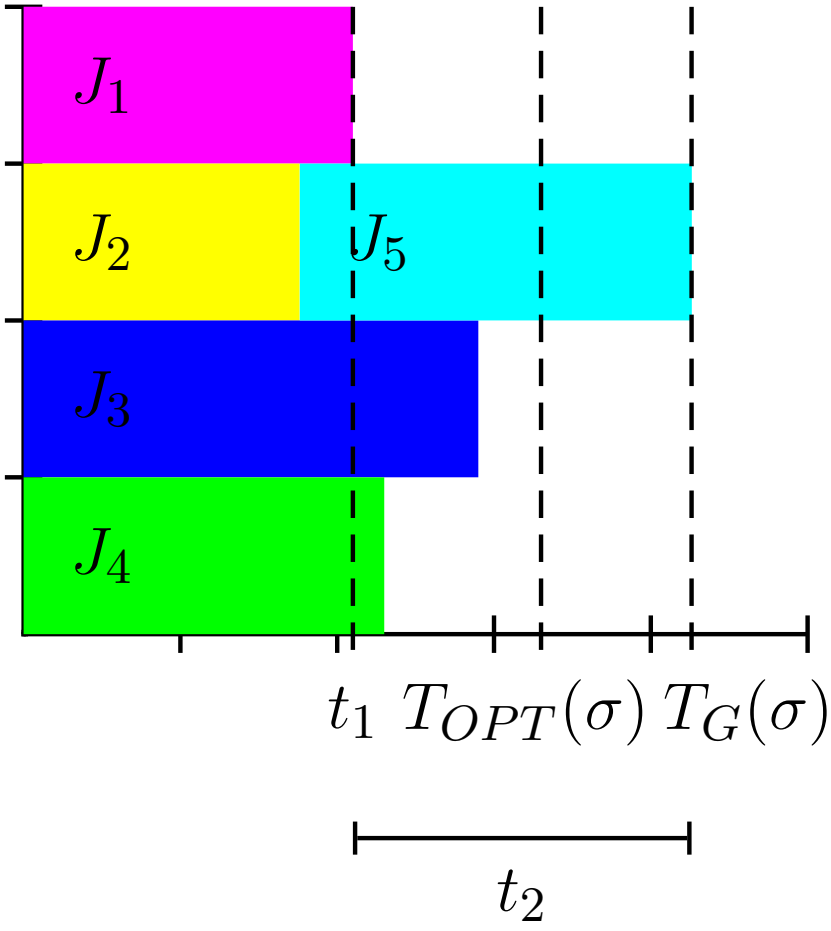


$T_G(\sigma)$ työjonon valmistumisaika AHNE algoritmin skeduloimana.

$T_{OPT}(\sigma)$ optimaalinen valmistumisaika.

t_1 aika, johon asti kaikki koneet ovat käytössä. $t_2 = T_G(\sigma) - t_1$.

Todistus



$T_G(\sigma)$ työjonon valmistumisaika AHNE algoritmin skeduloimana.

$T_{OPT}(\sigma)$ optimaalinen valmistumisaika.

t_1 aika, johon asti kaikki koneet ovat käytössä. $t_2 = T_G(\sigma) - t_1$.

Osoitetaan, että

$$T_G(\sigma) \leq 2T_{OPT}(\sigma).$$

Todistus

Viimeiseksi valmistuva työ alkaa hetkellä $t \leq t_1$, joten sen kesto on vähintään t_2.

Huomataan $t_2 \leq \max p_k$ ja $t_1 \leq \frac{1}{m} \sum p_k$.

Selvästi $T_{OPT}(\sigma) \geq \max\{t_1, t_2\}$, joten

$$T_G(\sigma) = t_1 + t_2 \leq 2 \max\{t_1, t_2\} \leq 2T_{OPT}(\sigma)$$

Parempia ratkaisuja

AHNE algoritmi pyrkii pitämään kaikkien koneiden rasituksen mahdollisimman tasaisena. Tämä tuottaa ongelmia, kun ajettavaksi tulee pitkä työ. Algoritmi **IMBALANCE** varautuu uusiin töihin kevyesti kuormitetuilla koneilla.

IMBALANCE: Olkoon $\alpha = 1.945$, h_i i:ntenä pysähtyvän koneen pysähtymisaika. Olkoon A_i $i-1$:n aikaisimmin pysähtyvien koneiden keskimääräinen pysähtymisaika ja $A_0 = \infty$. Skeduloi työ J_k viimeiseksi pysähtyvälle koneelle j siten että $h_j + p_k \leq \alpha A_j$

Algoritmi **IMBALANCE** on 1.945 kilpailukykyinen. Paras tunnettu deterministinen algoritmi on 1.9201 kilpailukykyinen ja paras satunnaisalgoritmi on 1.916 kilpailukykyinen [Albers, 2002]. Kilpailukyvyn alaraja deterministiselle algoritmille on 1.852 [Albers, 1999]

Rajoitetumpia skeduloitiongelmia

Identtiset koneet, rajoitettu käyttö m identtistä konetta mutta uusia töitä annetaan vain rajoitetulle joukolle koneista. AHNE algoritmi $O(\log m)$ kilpailukykyinen.

Samankaltaiset koneet (related machines) Konella on nopeus s_i. Työn J_k ajoaika on p_k/s_i. AHNE algoritmi $O(\log m)$ kilpailukykyinen. Olemassa myös 8 kilpailukykyinen algoritmi.

Erilaiset koneet (unrelated machines) Työn J_k ajo koneella i kestää $p_{k,i}$. AHNE on m kilpailukykyinen. Olemassa myös $O(\log m)$ kilpailukykyinen algoritmi.

Rasituksen tasaus

Työllä J_k on paino $w(k)$ ja työn kesto on tuntematon. Koneen i rasitus hetkellä t on $l_i(t)$.
Tavoitteena on minimoida työsarjan $\sigma = J_1, \ldots, J_n$ ajosta seuraava maksimaalinen rasitus.

Identtisten koneiden ja rajoitetun käytön tapauksessa, kilpailukyvyllä on $\Omega(\sqrt{m})$ alaraja, joka saavutetaan esim **ROBIN HOOD** algoritmilla.

ROBIN HOOD

ROBIN HOOD: Ylläpidetään arviota $L \leq OPT$. Hetkellä t, kone i on *rikas*, jos $l_i(t) \geq \sqrt{m}L$. Muuten kone on *köyhä*. Työn J_k saapuessa, päivitetään

$$L \leftarrow \max\{L, w(k), \frac{1}{m}(w(k) + \sum_{i=1}^{m} l_i(t))\}.$$

Jos mahdollista, skeduloi J_k köyhälle koneelle. Muuten skeduloi se rikkaalle, joka rikastui viimeksi.

ROBIN HOOD

ROBIN HOOD: Ylläpidetään arviota $L \leq OPT$. Hetkellä t, kone i on *rikas*, jos $l_i(t) \geq \sqrt{m}L$. Muuten kone on *köyhä*. Työn J_k saapuessa, päivitetään

$$L \leftarrow \max\{L, w(k), \frac{1}{m}(w(k) + \sum_{i=1}^{m} l_i(t))\}.$$

Jos mahdollista, skeduloi J_k köyhälle koneelle. Muuten skeduloi se rikkaalle, joka rikastui viimeksi.

Lause 29 *ROBIN HOOD on $O(\sqrt{m})$ kilpailukykyinen.*

ROBIN HOOD

Lemma 5 *Korkeintaan* $\lceil \sqrt{m} \rceil$ *konetta voi olla kerrallaan rikkaita.*

Muuten $\lceil \sqrt{m} \rceil \sqrt{m} L \geq mL$, vaikka $L \leq \frac{1}{m} \sum_{i=1}^{m} l_i(t)$.

Lemma 6 *Aina* $L \leq OPT$.

Induktiolla ajossa olevien töiden suhteen kun L muuttuu.
Huomataan $w(k) \leq OPT$ ja $\frac{1}{m} \sum_{i=1}^{m} l_i(t) \leq OPT$.

ROBIN HOOD

Lause 29 *ROBIN HOOD on* $2\lceil\sqrt{m}\rceil$ *kilpailukykyinen.*

Tarkastellaan ajan hetkeä t. Osoitetaan, että $l_i(t) \leq \lceil\sqrt{m}\rceil(L + OPT)$.

ROBIN HOOD

Lause 29 *ROBIN HOOD on* $2\lceil\sqrt{m}\rceil$ *kilpailukykyinen.*

Tarkastellaan ajan hetkeä t. Osoitetaan, että $l_i(t) \leq \lceil\sqrt{m}\rceil(L + OPT)$. Jos kone i on köyhä, väite on selvä.

ROBIN HOOD

Lause 29 *ROBIN HOOD on* $2\lceil\sqrt{m}\rceil$ *kilpailukykyinen.*

Tarkastellaan ajan hetkeä t. Osoitetaan, että $l_i(t) \leq \lceil\sqrt{m}\rceil(L + OPT)$. Jos kone i on köyhä, väite on selvä.

Olkoon t_0 hetki, jolloin kone i tuli viimeksi rikkaaksi. Olkoon $M(t_0)$ koneet, jotka ovat rikkaita hetkellä t ja jotka tulivat rikkaiksi viimeistään hetkellä t_0. Olkoon S ne käynnissä olevat työt, jotka skeduloitiin koneelle i hetken t_0 jälkeen.

Joukon S työt voitiin skeduloida vain joukon $M(t_0)$ koneille sillä muuten ne olisi skeduloitu köyhille koneille eikä koneelle i. Selvästi $OPT \geq \frac{1}{|M(t_0)|}\sum_{J_k \in S} w(k)$.

ROBIN HOOD

Olkoon J_q työ, joka aiheutti koneen i rikastumisen. Oletetaan aluksi, että $|M(t_0)| \leq \lceil \sqrt{m} \rceil - 1$. Nyt

$$l_i(t) \leq \lceil \sqrt{m} \rceil L + w(q) + \sum_{J_k \in S} w(k) \leq \lceil \sqrt{m} \rceil (L + OPT).$$

Viimeinen epäyhtälö seuraa siitä, että $w(q) \leq OPT$ ja $\sum w(k) \leq |M(t_0)|OPT$.

Jos $|M(t_0)| = \lceil \sqrt{m} \rceil$ niin $l_i(t_0) = \sqrt{m}L$. Muuten jouduttaisiin ristiriitaan, että yhteisrasitus olisi suurempi kuin mL vaikka L on vähintään keskimääräinen rasitus konetta kohti. Tästä seuraa

$$l_i(t) \leq \sqrt{m}L + \sum_{J_k \in S} w(k) \leq \lceil \sqrt{m} \rceil (L + OPT).$$

Robotin navigointi

- Navigaatio ongelmassa on tarkoitus löytää reitti lähteestä s kohteeseen t.
- Kartoitusongelmassa on tarkoitus rakentaa koko ympäristön kartta.
- Molemmissa tapauksissa tavoitteena on minimoida kuljettu matka.
- Navigaatioalgoritmi on c kilpailukykyinen, jos robotin kulkema matka on korkeintaan c kertaa lyhimmän s:n ja t:n välisen polun pituus.

Sokea robotti suoralla

Tarkastellaan robotin navigointia 1 ulotteisessa avaruudessa. Robotti on aluksi origossa $s = 0$ eikä tiedä kummassa suunnassa t sijaitsee. Robotti havaitsee vain saavuttaneensa t:n.

Optimaalinen strategia on edetä kerralla i origosta matka $f(i) = (-2)^i$. Kun etäisyys $t - s = n$, saadaan

$$2 \sum_{i=1}^{\lfloor \log n \rfloor + 1} 2^i + n \leq 9n.$$

Tarkastellaan seuraavaksi pelkästään tuntoaistia käyttävää robottia 2 ulotteisessa avaruudessa.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Lähteen ja kohteen etäisyys on n. Kohde sijaitsee äärettömän leveällä seinällä.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Robotti etenee kohti t:tä. Etäisyyden 1 jälkeen eteen tulee este.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Kaikkien esteiden leveys on $\frac{1}{2}$ ja pituus $2n$

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Jossain vaiheessa robotti kiertää esteen ja etenee oikealle.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Kun robotti on edennyt matkan 1 oikealle, sen eteen asetetaan uusi este.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Näin asetetaan n estettä.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Näin asetetaan n estettä.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Näin asetetaan n estettä.

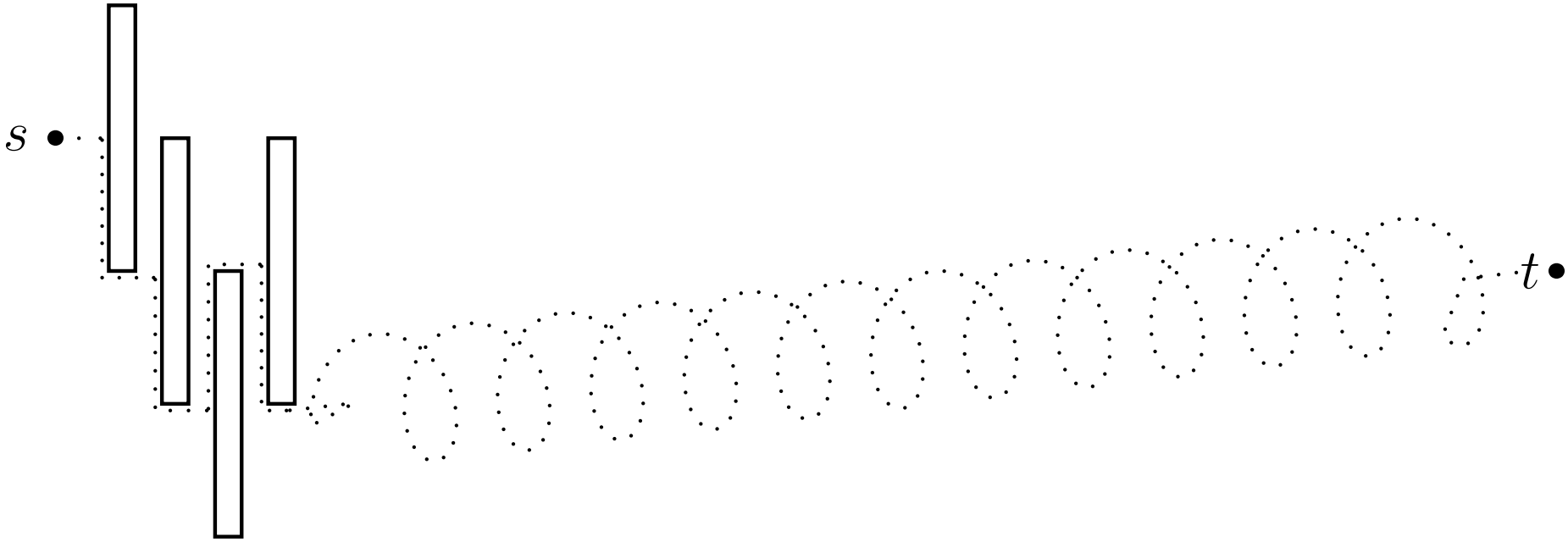
Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*

Näin asetetaan n estettä.

Alaraja robottinavigoinnille

Lause 30: *Jokaisen deterministisen online navigointialgoritmin kilpailukyky on vähintään $\Omega(\sqrt{n})$, mikäli avaruudessa on n suorakulmaista ja akselien suuntaista estettä.*



Robotti joutuu kulkemaan vähintään matkan $O(n^2)$.

Alaraja robottinavigoinnille

Koska avaruudessa on vain n estettä, jollain y–koordinaatin arvolla, joka on korkeintaan $n^{3/2}$ sivussa s:stä, edessä on vain $\sqrt{n}$ estettä. Tämä nähdään siitä, että $2n$ levyisiä viipaleita voidaan peittää $\sqrt{n}$ esteellä korkeintaan $n/\sqrt{n} = \sqrt{n}$ kappaletta. Yhteensä peitetty alue siis on $2n\sqrt{n} = 2n^{3/2}$.

Yhä edessä olevien $\sqrt{n}$ esteen kiertäminen vaatii korkeintaan $n \cdot \sqrt{n} = n^{3/2}$ matkan. Näin ollen mikä tahansa deterministinen online algoritmi on $\Omega(\sqrt{n})$ huonompi kuin paras offline algoritmi.

Robottinavigointi

Blum et.al. kehitti rajan $O(\sqrt{n})$ saavuttavan deterministisen online navigointialgoritmin vuonna 1991. Samana vuonna Papadimitriou et.al. todisti itse alarajan. Bergman esitti vuonna 1996 satunnaisalgoritmin, joka saavuttaa välinpitämätöntä vastustajaa kohti kilpailukyvyn $O(n^{4/9} \log n)$.

Harjoitustehtävät

1. Todista että erilaisilla koneilla AHNE algoritmin kilpailukyky on $\Omega(n)$.
2. Todista että erilaisilla koneilla AHNE algoritmin kilpailukyky on $O(n)$.

Kirjallisuus

References

[Albers, 1999] Albers, S. (1999). Better bounds for online scheduling. *SIAM Journal on Computing*, 29(2):459–473.

[Albers, 2002] Albers, S. (2002). On randomized online scheduling. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 134 – 143.