

JOHDATUS TEKOÄLYYN

TEEMU ROOS

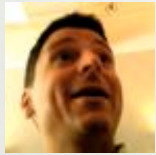


HELSINGIN YLIOPISTO

KURSSIN PERUSTIEDOT

- * VALINNAINEN AINEOPINTOTASOINEN KURSSI, 4 OP
- * PERIODI 1: 5.9.2012-11.10.2012 (6 VIIKKOA+KOE)
- * LUENNOT (B123, LINUS TORVALDS -AUDITORIO):
TO 10-12, PE 12-14
- * LASKUHARJOITUKSET (B222):
RYHMÄ 1: MA 12-14 (TEEMU)
RYHMÄ 2: MA 14-16 (KALLE)
RYHMÄ 3: KE 10-12 (HANNU)
RYHMÄ 4: KE 12-14 (KALLE)
- * KURSSIKOE PE 18.10.2012 KLO 9-12 A111

TIIMI



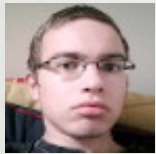
TEEMU.ROOS@CS.HELSENKI.FI

IRC: teemuroos HUONE: A322



HANNU KÄRNÄ

IRC: hakarna



KALLE VIIRI

IRC: EKK

IRC: #johtek


ESITIIETOVAATIMUKSET

- * TIETORAKENTEET-KURSSI
- * JOHDATUS YLIOPISTOMATEMATIIKKAAN -KURSSI
- * TODENNÄKÖISYYSLASKENNAN KURSSISTA HYÖTYÄ
- * OHJELMOINTITAITO
- * KIELI VAPAA.
JAVAAN OHJAUSTA.

MITÄ PITÄÄ TEHDÄ?

- * LUENNOILLA EI OLE PAKKO ISTUA
- * KURSSIKIRJAA EI OLE -- MATERIAALI KURSSIN SIVULLA
 - KURSSIMONISTE
 - LUENTOKALVOT (SIS. LINKKEJÄ)
- * LASKUHARJOITUKSET MAX 20 PISTETTÄ
- * KURSSIKOE MAX 40 PISTETTÄ
- * HYVÄKSYMISRAJA N. 30 PISTETTÄ

VIELÄ PARI JUTTUA

- * KURSSI ERILAINEN KUIN VIIME VUOSINA:
 - ERI LUENNOIJA (NYT 3. VUOTTA)
 - ERI PAINOTUKSET
- * RAKENTAVA KRITIIKKI TERVETULLUTTA!
- * TAVOITE: 100% LÄPÄISEE 
- * TYÖMÄÄRÄ:
YHTEENSÄ N. 100 TUNTIA TAI 15 TUNTIA VIIKOSSA
- * "NO PAIN, NO GAIN!"
- * "ALL WORK AND NO PLAY MAKES JACK A DULL BOY"

AIHEITA

1. MITÄ ON TEKOÄLY? HISTORIA JA FILOSOFIA

2. PELITJA ETSINTÄ

"GOFAI"

3. ~~LOGIIKKA (OHJELMOINTI)~~

4. ROBOTIIKKA JA SIGNAALINKÄSITTELY "MODERN AI"

5. KONEOPPIMINEN JA PÄÄTTELY EPÄVARMUUDEN
VALLITESSA

6. ~~LUONNOLLISEN KIELEN KÄSITTELY~~

KESKUSTELUA

* TEKOÄLY KULTTUURISSA

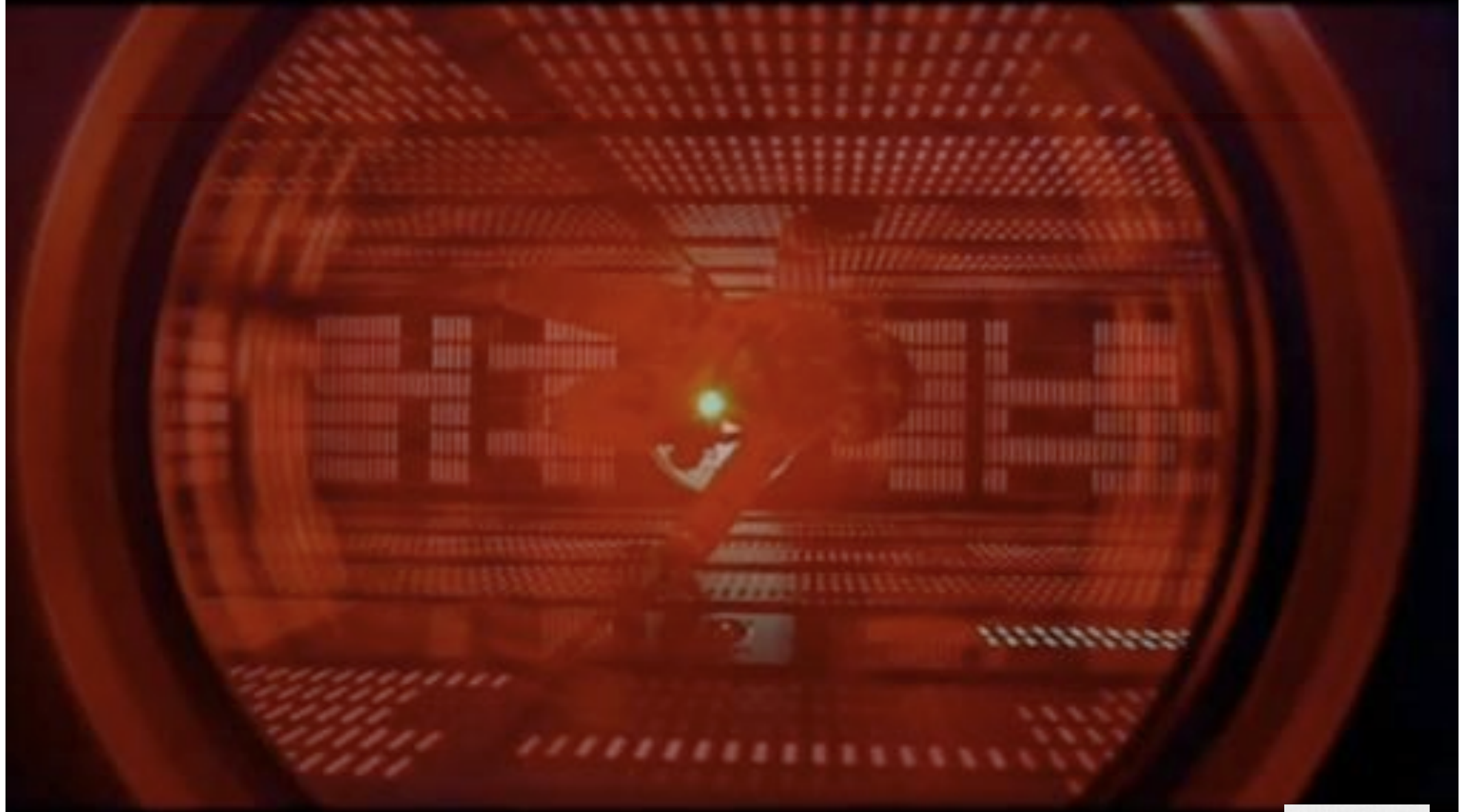
* _____ SKYNET _____

* _____ R2D2 _____

* _____ GLADOS _____

* _____ HAL9000 _____

* _____ EDUSKUNTA _____

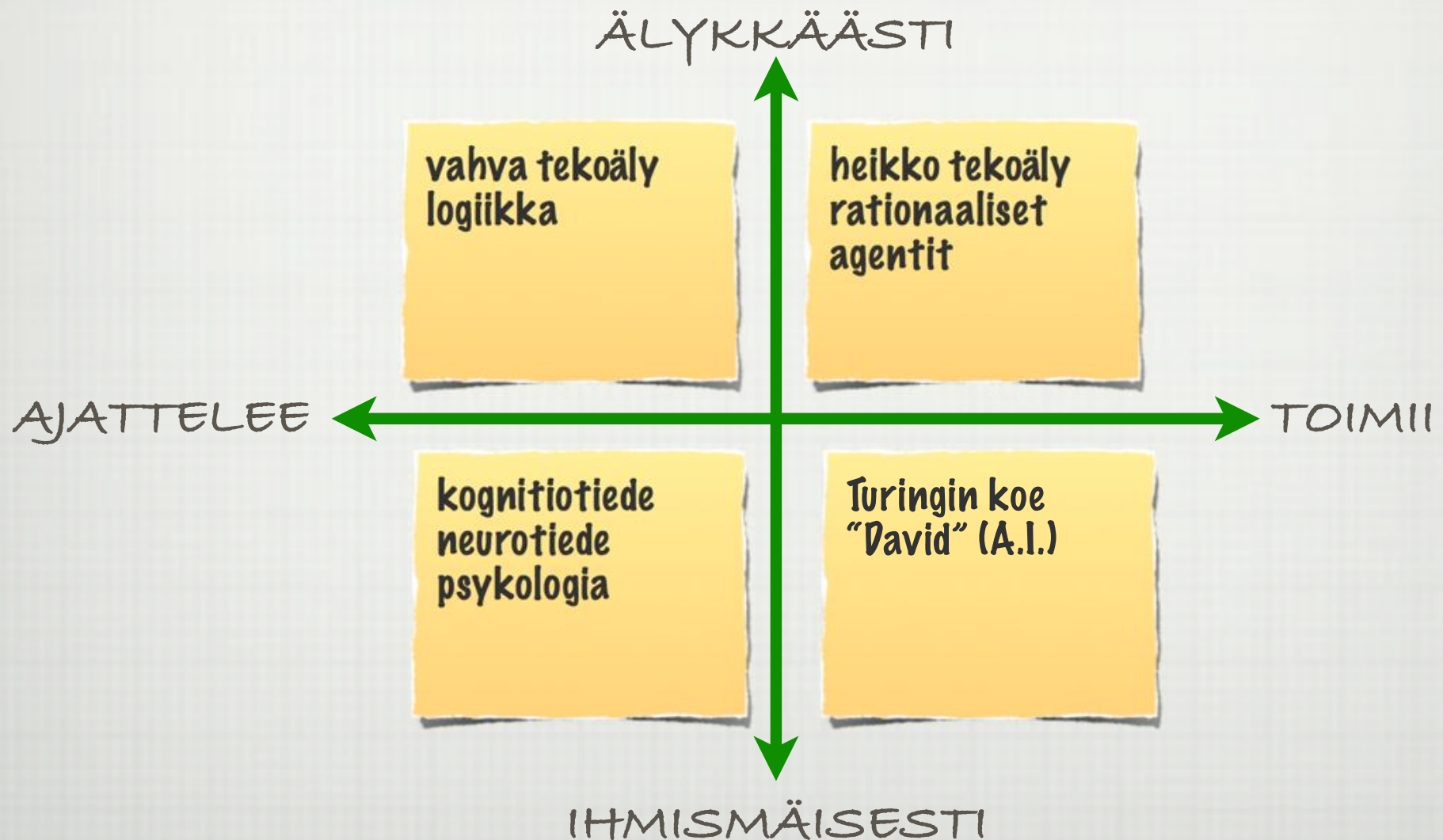


Deactivati

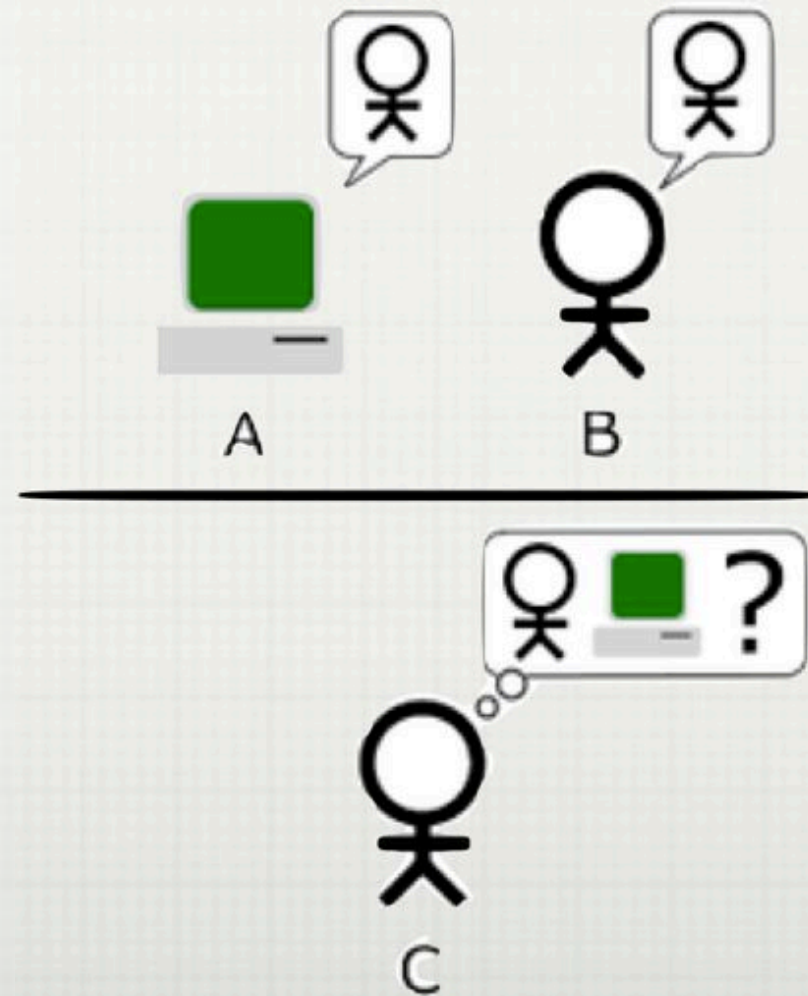
KESKUSTELUA

- * MITÄ KAIKKEA HAL OSAA?
- * MITÄ NÄISTÄ EI OSATA VIELÄ TOTEUTTAA?
- * ONKO HALILLA TIETOISUUS?

TEKOÄLYN FILOSOFIAA



TOIMII IHMISMÄISESTI: TURINGIN TESTI





Are you good?

Artificial Intelligence
A Modern Approach

KIINALAINEN HUONE

- * VOIKO TOIMIA
ÄLYKKÄÄSTI ILMAN
ETTÄ AJATTELEE?
- * TIETOISUUS?



MITÄ TEKOÄLY OIKEASTI ON?

KONENÄKÖ

REITINOPTIMOINTI

KONEOPPIMINEN

TIEDONHAKU

KONEKÄÄNNÖS

SUOSITTELU

NLP

PUHE

PELIT

LOGIIKKA

TIEDON LOUHINTA

Recommended for You

**Networks, Crowds, and Markets:
Reasoning About a Highly Connected
World**

David Easley (Author), Jon
Bernstein (Author)

Price: **\$39.83**

Used & new from \$36.85

Add to Cart

Add to Wish List

Because you recently viewed...

MITÄ TEKOÄLY OIKEASTI ON?

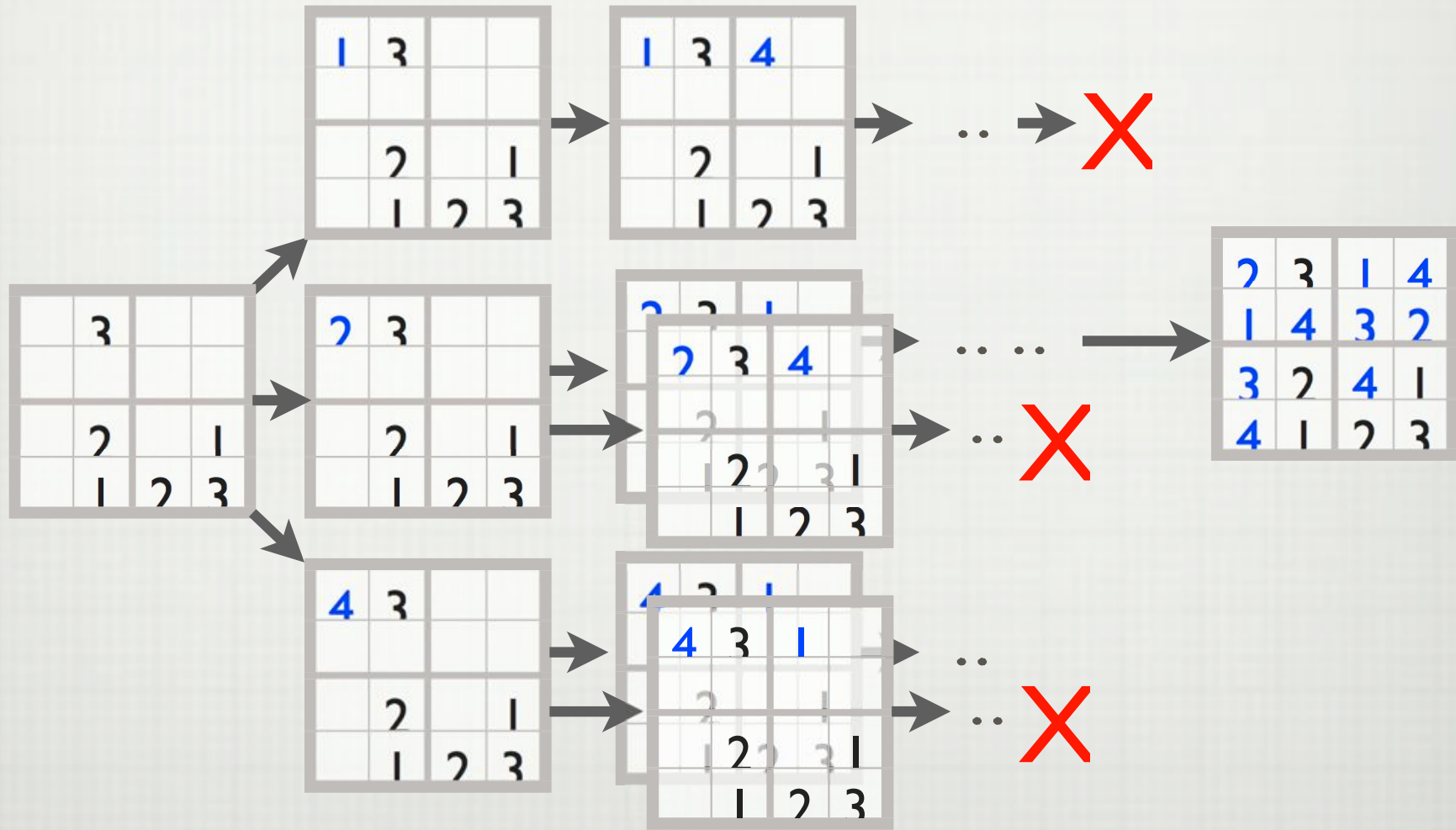
The screenshot shows the website for the 23rd International Joint Conference on Artificial Intelligence (IJCAI-13). The header includes the conference logo and title, along with the dates and location: August 3-8, 2013, Beijing, China, at the Beijing International Convention Center (BICC). A navigation menu includes links for HOME, CALLS, PROGRAM, COMMITTEES, REGISTRATION, ATTENDING, SPONSOR ACCOUNTS, STUDENTS MONITORING, and BILLING. A search bar is also present.

The main content area displays the daily program for Tuesday, August 6th, 2013 (Day 1). The program is organized into a table with time slots, room numbers, session titles, and speakers.

Time	Room	Session Title	Speakers	
08:30 - 09:30	Hal #2	Welcome address and program overview	Francesca Rossi	
09:30 - 09:45	Hal #2	Invited talk: Computational Disaster Management	Pascal Van Hentenryck	
09:45 - 10:15	Chair: Francesca Rossi			
Coffee Break				
10:15 - 11:00	Hal #2	Invited talk: Computational Perspectives on Social Phenomena at Global Scales	Jon Kleinberg	
11:15 - 12:00	Hal #2	Chair: Kevin Leyton-Brown		
11:15 - 12:00	Chair: Craig Knoblock, Introduced by Raymond J. Mooney	UJCAI-13 Computers and Thought Award: Towards Large-Scale Visual Recognition and Search	Kristen Grauman	
Lunch				
Award papers				
	UJCAI 2013 distinguished paper	Bayesian Optimization in High Dimensions: Ziyi Wang, Mounir Zghal, Frank Hutter, via Random Embeddings	David Matheson, Nando de Freitas	
	Room 201CD UJCAI 2013 distinguished paper	Maximizing Flexibility in Simple Temporal Networks	Bob Hueman, Tomas Klos, Michel Wilson, Coen Wittenberg	
	Session Chair: Francesca Rossi	UJCAI-JAIR 2013 best paper award	Adaptive Submodularity: Theory and Applications in Active Learning and Stochastic Optimization	Seriel Gokovin, Andreas Krause
	2012 ECCV	Bayesian Submodular Maximization		

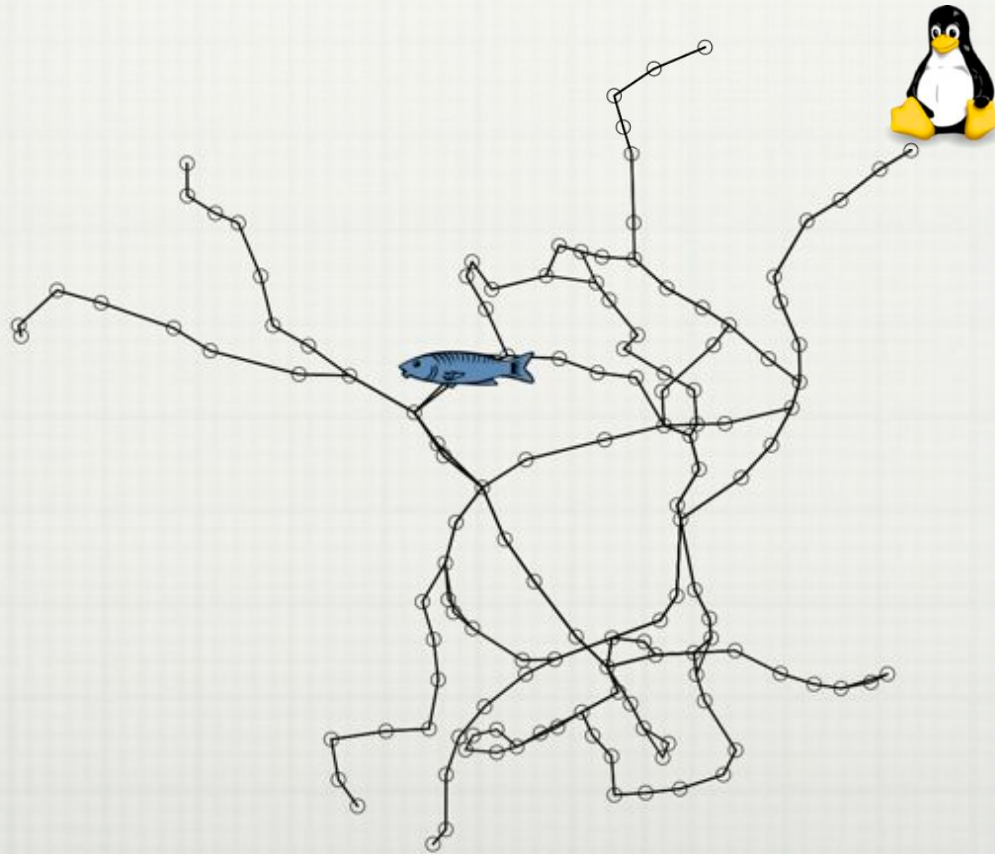
On the left side of the page, there is a 'PROGRAM' sidebar with links to 'Conference at a glance', 'Technical Program', 'Workshops', 'Tutorials', 'Invited talks', 'Doctoral Consortium', 'IJCAI-13 Awards', 'Angry Birds AI Competition', 'Robot Competition and Exhibition', 'Video competition', and 'Special meetings'. Below this is an 'ACCEPTED PAPERS' section with a link to 'Main and ACS track'.

ETSINTÄ

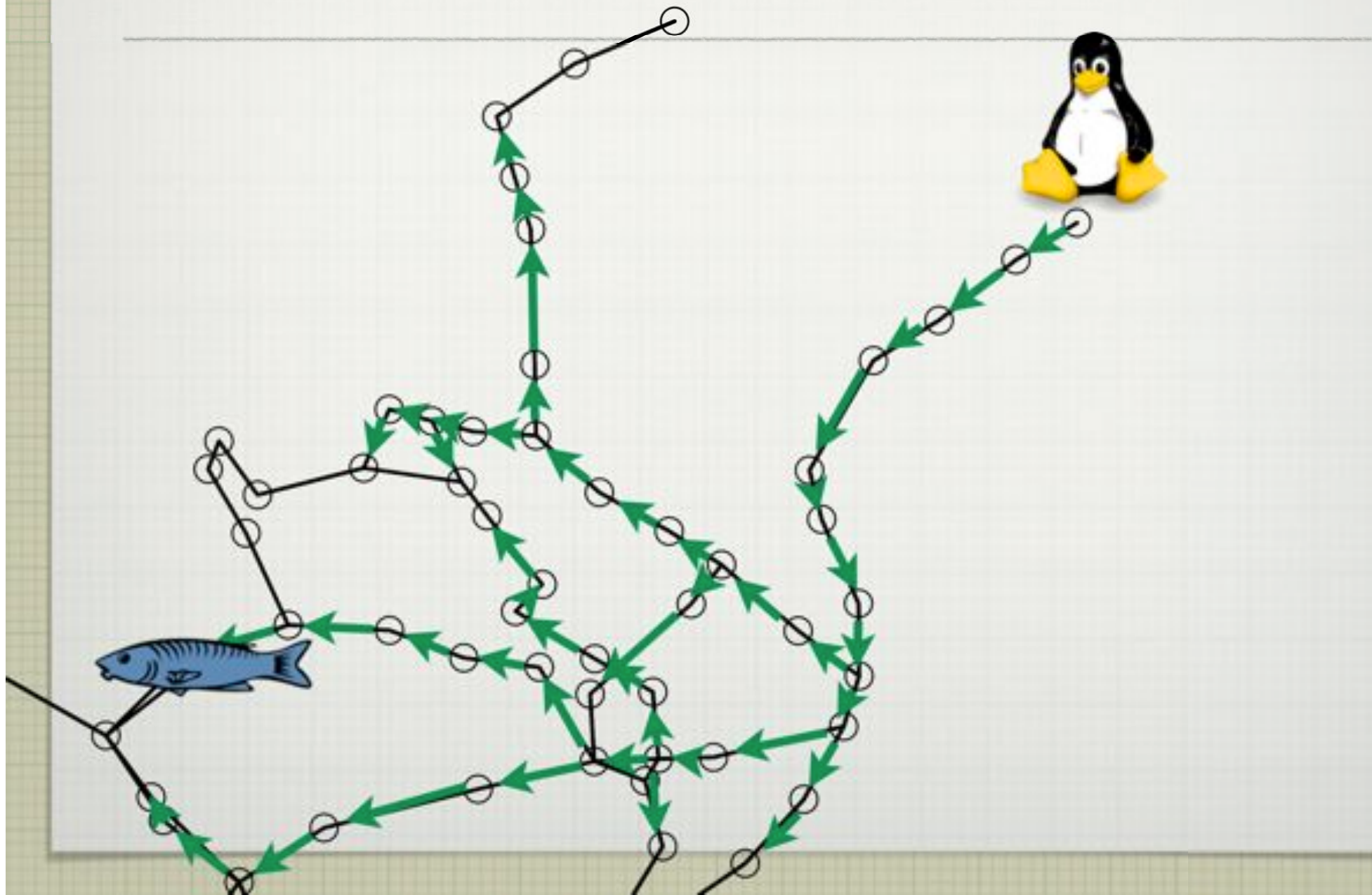


ETSINTÄ

LEVEYSSUUNTAINEN HAKU

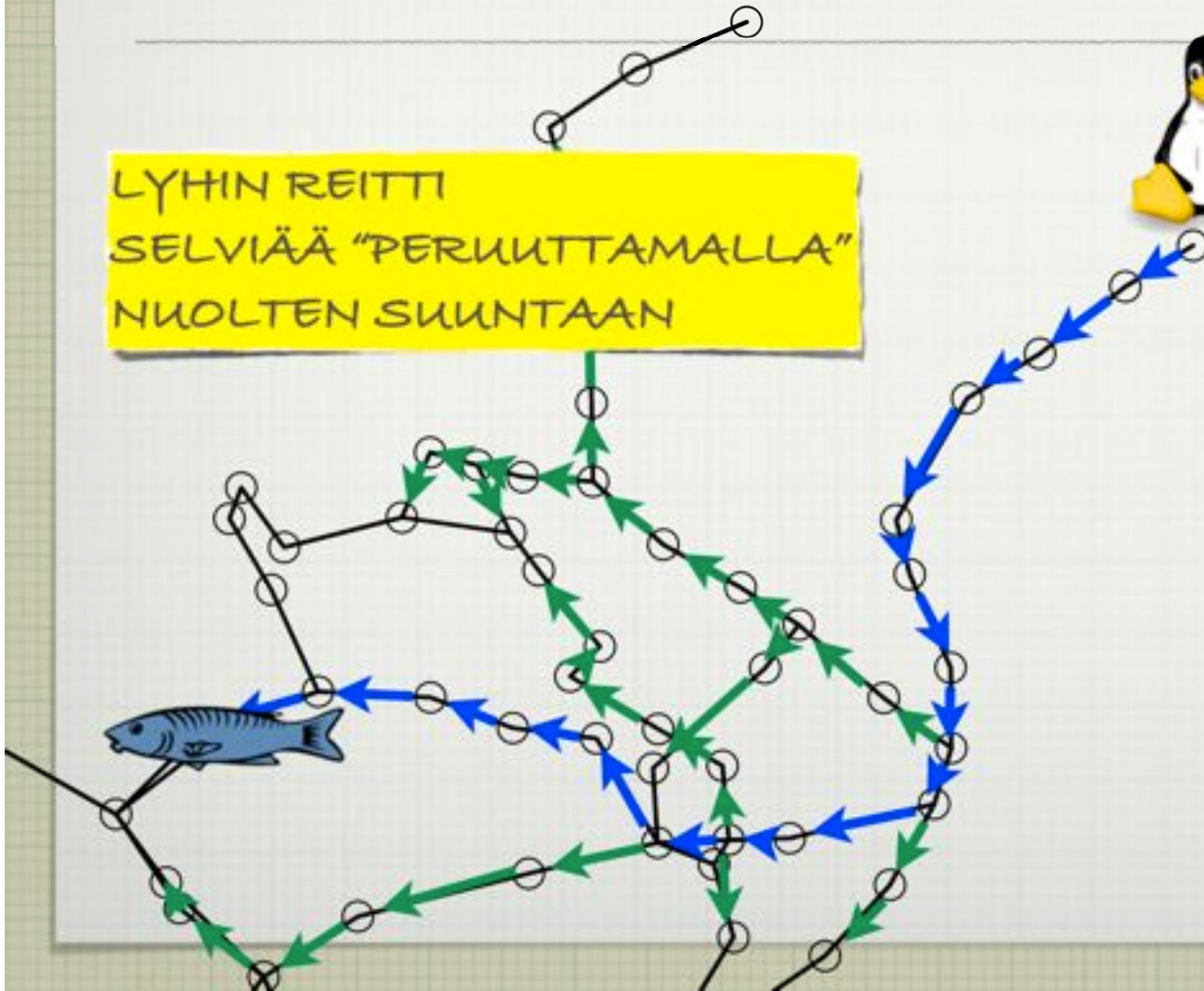


ETSINTÄ



ETSINTÄ

LYHIN REITTI
SELVIÄÄ "PERUUTTAMALLA"
NUOLTEN SUUNTAAN



ETSINTÄ

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

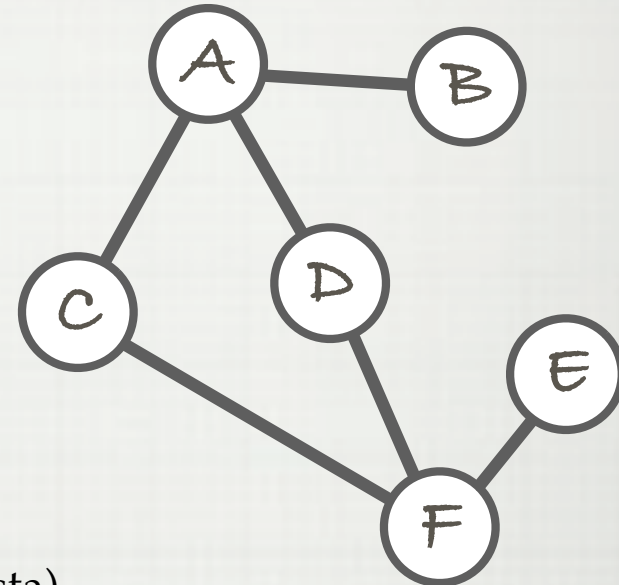
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

[A]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

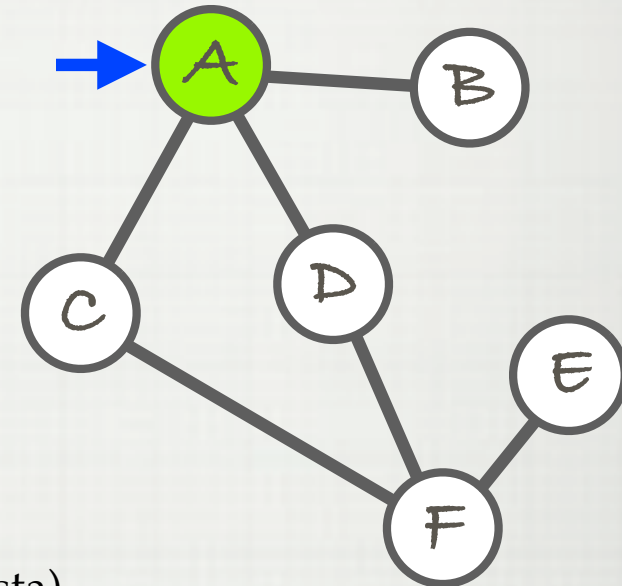
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista - Käsitellyt - Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ



ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

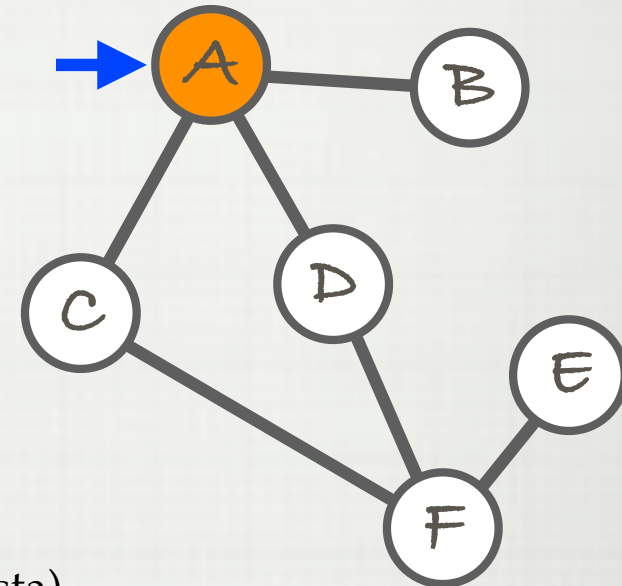
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

[B,C,D]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

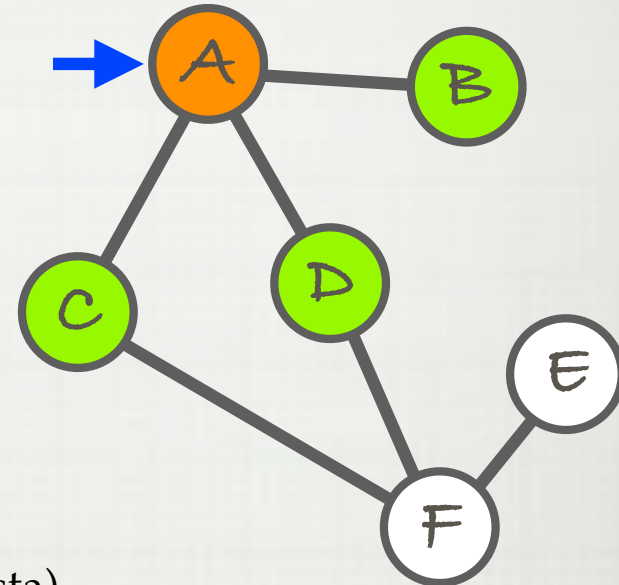
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

[C,D]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

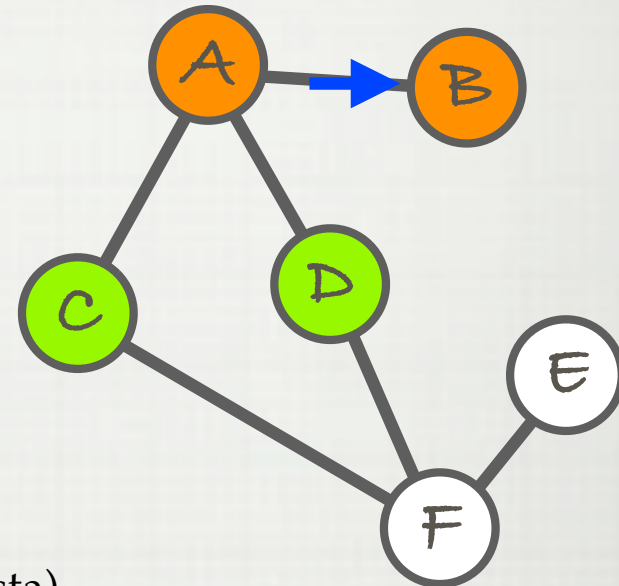
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

[D]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

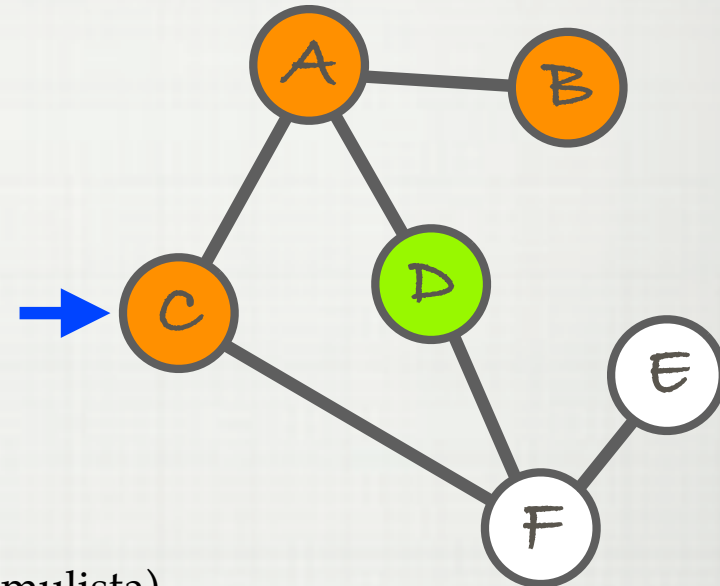
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

[D,F]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

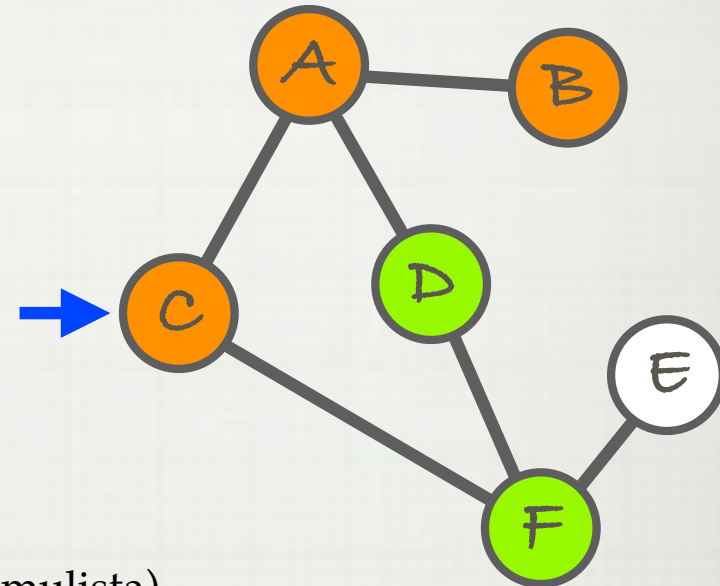
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista - Käsitellyt - Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

2. ETSINTÄ JA PELIT

[F]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

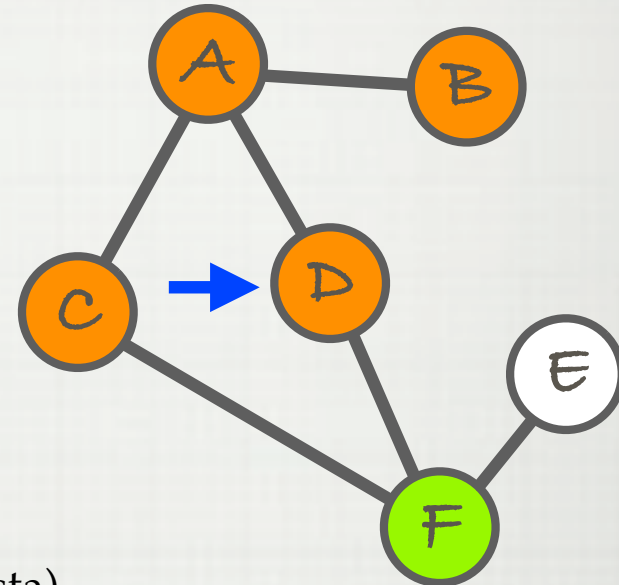
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ



ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

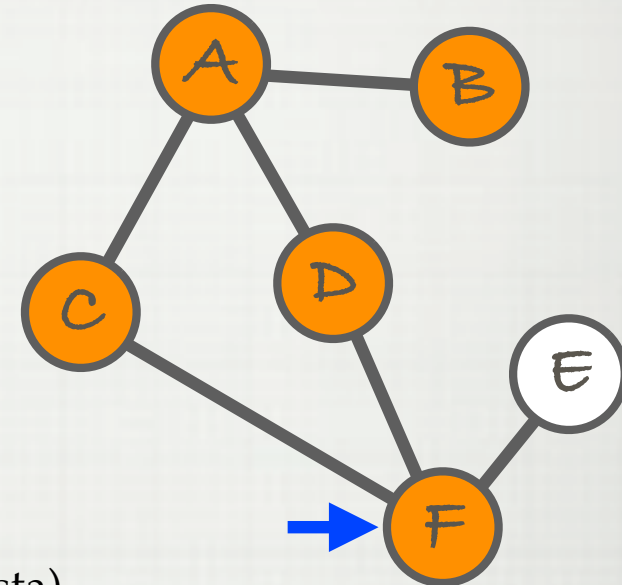
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

[E]

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

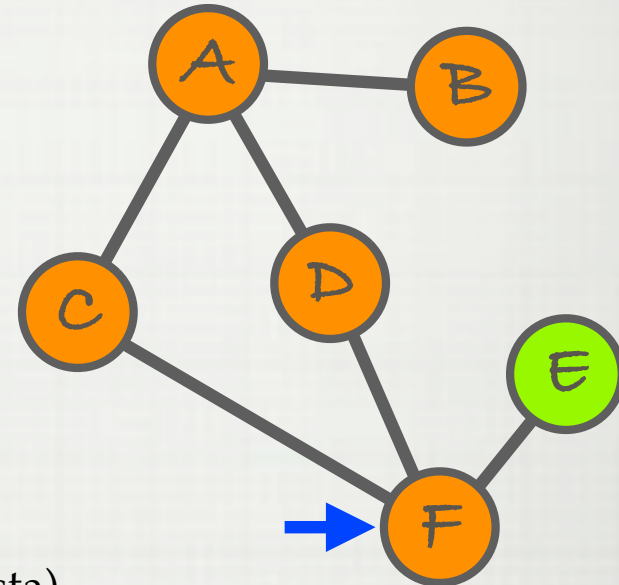
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ



ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

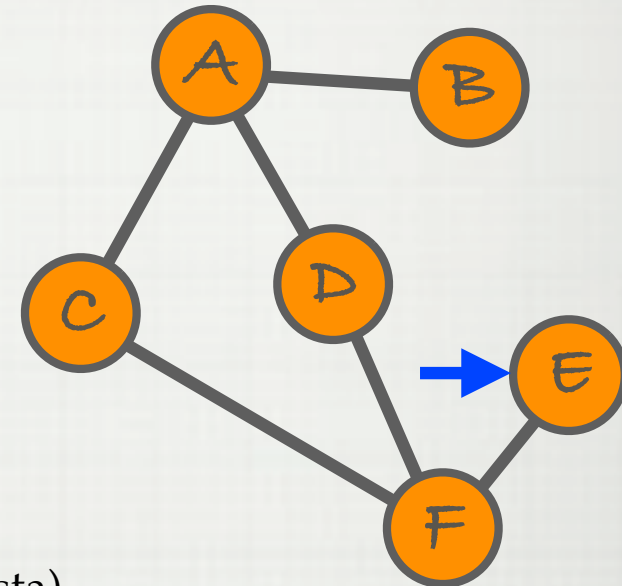
 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")



LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

Uudet = Naapurilista – Käsitellyt – Solmulista

return(PERÄKKÄIN(Solmulista, Uudet))

LISÄÄ([a,f],[d]) => [d,a,f]

ETSINTÄ

ETSINTÄ(Alkusolmu)

```
Solmulista = [Alkusolmu]
Käsitellyt = [ ]
while Solmulista not empty
  Solmu = EKA(Solmulista)
  Solmulista = LOPUT(Solmulista)
  if Solmu not in Käsitellyt
    Käsitellyt = Käsitellyt + [Solmu]
    if MAALI(Solmu) return("ratkaisu", Solmu)
    Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)
  end if
end while
return("ei ratkaisua")
```

JONO ("FIRST-IN-FIRST-OUT")

LISÄÄ(Naapurit, Solmulista)

LEVEYSSUUNTAINEN HAKU:

```
Uudet = Naapurilista - Käsitellyt - Solmulista
return(PERÄKKÄIN(Solmulista, Uudet))
# LISÄÄ([a,f],[d]) => [d,a,f]
```


ETSINTÄ

ETSINTÄ(Alkusolmu)

```
Solmulista = [Alkusolmu]
Käsitellyt = [ ]
while Solmulista not empty
  Solmu = EKA(Solmulista)
  Solmulista = LOPUT(Solmulista)
  if Solmu not in Käsitellyt
    Käsitellyt = Käsitellyt + [Solmu]
    if MAALI(Solmu) return("ratkaisu", Solmu)
    Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)
  end if
end while
return("ei ratkaisua")
```

PINO ("LAST-IN-FIRST-OUT")

LISÄÄ(Naapurit, Solmulista)

SYVYYSSUUNTAINEN HAKU:

```
Uudet = Naapurilista - Käsitellyt
return(PERÄKKÄIN(Uudet, Solmulista))
# LISÄÄ([a,f],[d]) => [a,f,d]
```

ETSINTÄ ONGELMANRATKAISUNA



ETSINTÄ ONGELMANRATKAISUNA

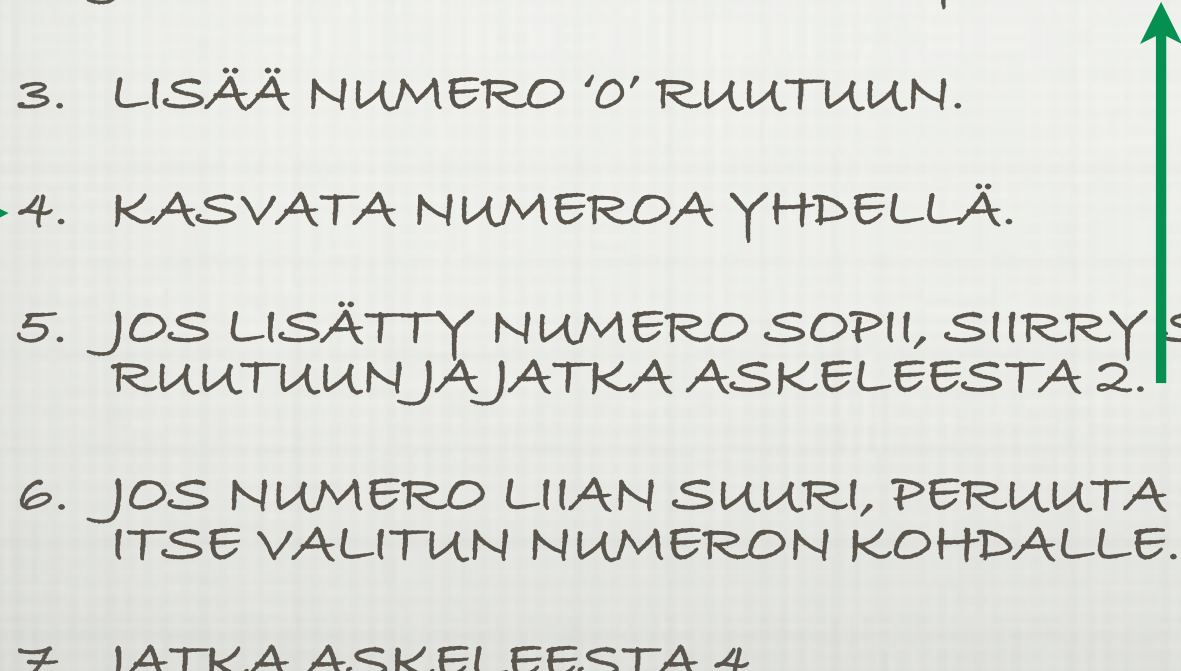


ETSINTÄ ONGELMANRATKAISUNA

- * KOLME KANNIBAALIA JA KOLME LÄHETYSSAARNAAJAA HALUAA YLITTÄÄ JOEN VENEELLÄ, JOHON MAHTUU VAIN KAKSI HENKILÖÄ.
- * JOS JOMMALLA KUMMALLA RANNALLA ON ENEMMÄN KANNIBAALIA KUIN LÄHETYSSAARNAAJIA (MUTTA KUITENKIN VÄHINTÄÄN YKSI LÄHETYSSAARNAAJA), KANNIBAALIT SYÖVÄT HEIDÄT.
- * MITEN JOKI SAADAAN YLITETTYÄ ILMAN, ETTÄ KETÄÄN SYÖDÄÄN?
- * VOIT KOKEILLA KLIKKAAMALLA TÄSTÄ.

SUDOKU

* YKSINKERTAINEN SUDOKU-ALGORITMI:

1. ALOITA VASEMMASTA YLÄKULMASTA.
 2. JOS RUUTU ANNETTU, SIIRRY SEURAAVAAN.
 3. LISÄÄ NUMERO '0' RUUTUUN.
 4. KASVATA NUMEROA YHDELLÄ.
 5. JOS LISÄTTY NUMERO SOPII, SIIRRY SEURAAVAAN RUUTUUN JA JATKA ASKELEESTA 2.
 6. JOS NUMERO LIIAN SUURI, PERUUTA EDELLISEN ITSE VALITUN NUMERON KOHDALLE.
 7. JATKA ASKELEESTA 4.
- 

SUDOKU

	3		
	2		1
	1	2	3

SUDOKU

1	3		
	2		1
	1	2	3

SUDOKU

1	3	1	
	2		1
	1	2	3

SUDOKU

1	3	2	
	2		1
	1	2	3

SUDOKU

1	3	3	
	2		1
	1	2	3

SUDOKU

1	3	4	
	2		1
	1	2	3

SUDOKU

1	3	4	2
	2		1
	1	2	3

SUDOKU

1	3	4	2
2			
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	4		
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	4	1	
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	4	1	?
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	4	3	
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	4	3	?
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	4	?	
	2		1
	1	2	3

SUDOKU

1	3	4	2
2	?		
	2		1
	1	2	3

SUDOKU

1	3	4	2
4			
	2		1
	1	2	3

SUDOKU

1	3	4	2
4	?		
	2		1
	1	2	3

SUDOKU

1	3	4	2
?			
	2		1
	1	2	3

SUDOKU

1	3	4	?
	2		1
	1	2	3

SUDOKU

1	3	?	
	2		1
	1	2	3

SUDOKU

2	3		
	2		1
	1	2	3

SUDOKU

2	3	1	
	2		1
	1	2	3

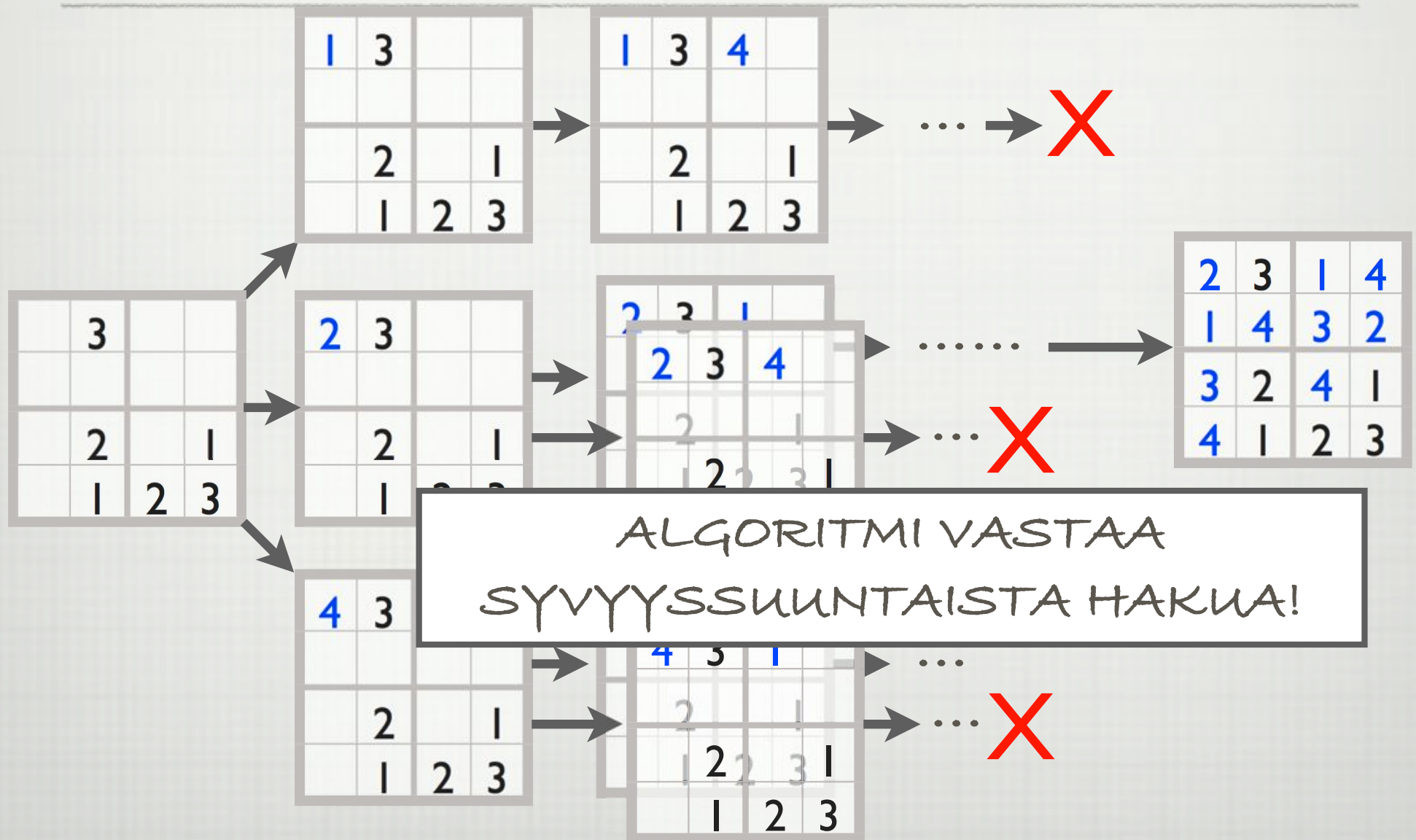
SUDOKU

2	3	1	4
	2		1
	1	2	3

SUDOKU

2	3	1	4
1	4	3	2
3	2	4	1
4	1	2	3

SUDOKU



PARAS-ENSIN-HAKU

ETSINTÄ(Alkusolmu)

Solmulista = [Alkusolmu]

Käsitellyt = []

while Solmulista not empty

 Solmu = EKA(Solmulista)

 Solmulista = LOPUT(Solmulista)

 if Solmu not in Käsitellyt

 Käsitellyt = Käsitellyt + [Solmu]

 if MAALI(Solmu) return("ratkaisu", Solmu)

 Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)

 end if

end while

return("ei ratkaisua")

LISÄÄ(Solmulista1, Solmulista2)

PARAS-ENSIN-HAKU:

return(JÄRJESTÄ(Solmulista1, Solmulista2))

[(a,5),(b,3),(c,1)], [(a,2),(c,3),(f,5)] => [(c,1),(a,2),(b,3),(f,5)]

HEURISTIIKAT

* **KUSTANNUSARVIO:** $f(N)$

- ARVIO LÄHTÖSOLMUSTA SOLMUN N KAUTTA
MAALISOLMUUN KULKEVAN POLUN KUSTANNUKSESTA

* **"HEURISTIIKKA":** $h(N)$

- ARVIO KUSTANNUKSESTA SOLMUSTA N MAALISOLMUUN

* **POLKUKUSTANNUS:** $g(N)$

- KUSTANNUS ALKUSOLMUSTA SOLMUN N
(RIIPPUU KULJETUSTA REITISTÄ)

$$f(N) = g(N) + h(N)$$

A*

ETSINTÄ(Alkusolmu)

```
Solmulista = [Alkusolmu]
Käsitellyt = [ ]
while Solmulista not empty
  Solmu = EKA(Solmulista)
  Solmulista = LOPUT(Solmulista)
  if Solmu not in Käsitellyt
    Käsitellyt = Käsitellyt + [Solmu]
    if MAALI(Solmu) return("ratkaisu", Solmu)
    Solmulista = LISÄÄ(NAAPURIT(Solmu),Solmulista)
  end if
end while
return("ei ratkaisua")
```

A*-HAKU: $f(N) = g(N) + h(N)$

LISÄÄ(Solmulista1, Solmulista2)

PARAS-ENSIN-HAKU:

return(JÄRJESTÄ(Solmulista1, Solmulista2))

[(a,5),(b,3),(c,1)], [(a,2),(c,3),(f,5)] => [(c,1),(a,2),(b,3),(f,5)]

A*

- * **OLETUS:** HEURISTIikka $h(N)$ ANTAA AINA ENINTÄÄN YHTÄ SUUREN ARVON KUIN TODELLINEN KUSTANNUS SOLMUSTA N MAALIIN.
- * TÄLLÖIN **A*** TUOTTA AINA OPTIMAALISEN RATKAISUN
- * TODISTUKSEN IDEA:
JONON EKAKSI EI VOI PÄÄSTÄ MAALISOLMU, JONKA POLKUKUSTANNUS ON SUUREMPI KUIN OPTIMAALISEN REITIN KUSTANNUS.
- * JOS HEURISTIikka "HYVÄ", SUURIMMASSA OSASSA HUONOJA SOLMUJA EI KÄYDÄ OLLENKAAN.

REITTIOPAS



Reittiopas

Taskuversio - På svenska - In English - Slangi - По-русски

Palautte - Ohjeet - FAQ

Reittiopas classic - Reittiopas API

HSL

Reittiopas

Omat lähdöt • Aikataulut • Linjaopas • Pyöräily ja kävely

Perushaku Tarkennettu haku

Mistä Kartta Tallenna Hakemisto

Mihin Kartta Tallenna Hakemisto

Kello 22 : 29 Lähtöaika
 Perillä

Pvm

	Ma	Ti	Ke	To	Pe	La	Su	☺
36	05	06	07	08	09	10	11	
Syyskuu 2011	12	13	14	15	16	17	18	
38	19	20	21	22	23	24	25	
39	26	27	28	29	30	01	02	
Lokakuu 40	03	04	05	06	07	08	09	

Hae

Poikkeusinfo



8.9.2011 17:35 - Sisäiset ja seutuläiinen myöhästyvät. Syy: ruuhka. Paikka: Mannerheimintie. Arvioitu kesto: 17:31 - 24:00.



8.9.2011 17:32 - Sisäiset ja seutulinjat myöhästyvät. Syy: ruuhka. Paikka: Mannerheimintie. Arvioitu kesto: 17:23 - 24:00.

[HSL /Poikkeusinfo](#) → [Mobiililaitteille](#) →

Liikennetiedotteet

- 8.9.2011 Linjalla 11 lisävuoroja Kissan yöhön 9.9.
- 8.9.2011 Martinkyläntien pysäkkien poikkeusjärjestelyt jatkuvat
- 7.9.2011 Bussille 14 reittimuutos Eirassa 12.9.
- 7.9.2011 Bussit ajavat poikkeusreittiä Keravan keskustassa 9. - 12.9.

Omat reitit

Ei omia reittejä (ohje omien reittien tallentamiseen).

Omat paikat

Ei omia paikkoja (ohje omien paikkojen tallentamiseen).

REITTIOPAS

- * TILA: (PYSÄKKI, KULUNUT AIKA)
- * KUSTANNUSARVIO: (MATKA-AIKA (MIN))
- * SIIRTYMÄT: (UUSI PYSÄKKI, VÄLIMATKA (MIN))
- * TEHTÄVÄ: ETSI NOPEIN REITTI PYSÄKILTÄ A PYSÄKILLE B
(EI KÄVELYÄ)
- * MENETELMÄ: A*-HAKU