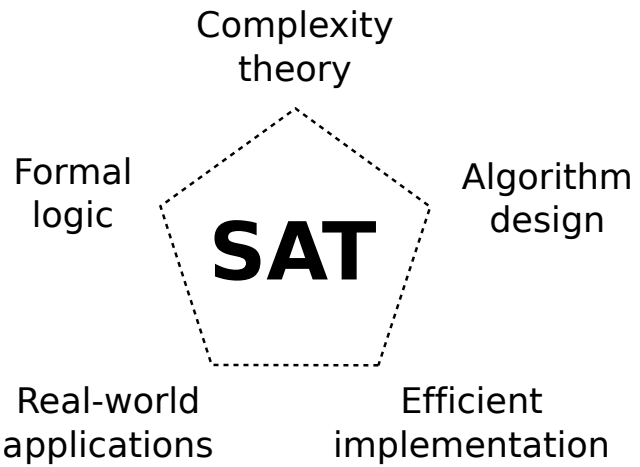
Moderni näkökulma logiikkaan tekoälytutkimuksessa

Matti Järvisalo

Constraint Reasoning and Optimization group
Tietojenkäsittelytieteen laitos
Helsingin yliopisto

29.2.2016



“Ei ainoastaan koneoppimista”

Aihepiirit modernissa tekoälytutkimuksessa

IJC**AI** & AA**AI**:

AI-tutkimuksen pääkonferenssit maailmanlaajuisesti

Konferenssien aihealueet:

Planning and Scheduling
Agent-based and Multi-agent systems
Combinatorial & Heuristic Search
Constraints & Satisfiability
Knowledge Representation, Reasoning and Logic
Machine Learning
Natural Language Processing
Robotics and Vision
Uncertainty in AI

Aihepiirit modernissa tekoälytutkimuksessa

IJC**AI** & AA**AI**:

AI-tutkimuksen pääkonferenssit maailmanlaajuisesti

Konferenssien aihealueet:

Planning and Scheduling
Agent-based and Multi-agent systems
Combinatorial & Heuristic Search
Constraints & Satisfiability
Knowledge Representation, Reasoning and Logic
Machine Learning
Natural Language Processing
Robotics and Vision
Uncertainty in AI

Aihepiirit modernissa tekoälytutkimuksessa

IJC**AI** & AA**AI**:

AI-tutkimuksen pääkonferenssit maailmanlaajuisesti

Konferenssien aihealueet:

Planning and Scheduling
Agent-based and Multi-agent systems
Combinatorial & Heuristic Search
Constraints & Satisfiability
Knowledge Representation, Reasoning and Logic
Machine Learning
Natural Language Processing
Robotics and Vision
Uncertainty in AI

Aihepiirit modernissa tekoälytutkimuksessa

IJC**AI** & AA**AI**:

AI-tutkimuksen pääkonferenssit maailmanlaajuisesti

Konferenssien aihealueet:

Planning and Scheduling
Agent-based and Multi-agent systems
Combinatorial & Heuristic Search
Constraints & Satisfiability
Knowledge Representation, Reasoning and Logic
Machine Learning
Natural Language Processing
Robotics and Vision
Uncertainty in AI

Aihepiirit modernissa tekoälytutkimuksessa

IJC**AI** & AA**AI**:

AI-tutkimuksen pääkonferenssit maailmanlaajuisesti

Konferenssien aihealueet:

Planning and Scheduling	*applikaatiot*
Agent-based and Multi-agent systems	*applikaatiot*
Combinatorial & Heuristic Search	*yhteydet*
Constraints & Satisfiability	
Knowledge Representation, Reasoning and Logic	
Machine Learning	*applikaatiot*
Natural Language Processing	*applikaatiot*
Robotics and Vision	*applikaatiot*
Uncertainty in AI	*applikaatiot*

Eksakteja ratkaisuja *vaikeisiin* ongelmiin

Tarve eksakteille ratkaisuille

Useissa tapauksissa hyödyllistä:

- *todistaa* että *ei ratkaisua*, tai
- löytää mahdollisimman *hyvä* ratkaisu.

Esimerkkejä:

- Find a **shortest** path/plan/execution/... to a goal state
  - Planning, model checking, ...
- Find a **smallest** explanation
  - Debugging, configuration, ...
- Find a **least resource-consuming** schedule
  - Scheduling, logistics, ...
- Find a **most probable** explanation (MAP)
  - Probabilistic inference, ...

Eksaktiuden tärkeys

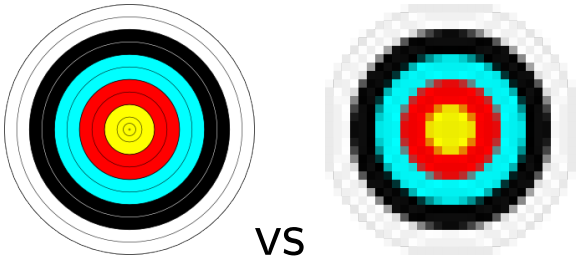
Annetaan periksi?

"The problem is NP-hard, so let's develop heuristics / approximation algorithms."

Ei!

Eksaktiuden hyötyjä:

- Resurssisäästöt
  - Aika, raha, ihmistyö
- Tarkkuus



Haaste: skaalautuminen *erittäin* isoihin ongelmiin

Eksaktiuden *välttämättömyys*

Ei vain *testausta* (software testing), vaan *oikeellisuuden todistamista*

- Logiikkapiirit, mikroprosessorit, ...
- Ohjelmistot, laiteajurit
- Sulautetut järjestelmät
- Turvallisuuskriittiset järjestelmät
  - Lentokoneet, raideliikenne, sairaalalaitteet, ...

Keskiössä: *automatisoitu looginen päättely*

SAT: lauselogiikan toteutumisongelma

- Propositional satisfiability, Boolean satisfiability

Eksaktiuden *välttämättömyys*

Ei vain *testausta* (software testing), vaan *oikeellisuuden todistamista*

- Logiikkapiirit, mikroprosessorit, ...
- Ohjelmistot, laiteajurit
- Sulautetut järjestelmät
- Turvallisuuskriittiset järjestelmät
  - Lentokoneet, raideliikenne, sairaalalaitteet, ...

Keskiössä: *automatisoitu looginen päättely*

SAT: lauselogiikan toteutumisongelma

- Propositional satisfiability, Boolean satisfiability

SAT-ongelma

Esimerkki: Luennoijan pukukoodiongelma

- Kolme rajoitetta jotka tulee toteuttaa:
  1. Selvästikään ei tule käyttää solmiota ilman paitaa.
  2. On epäkohteliasta olla sekä ilman solmiota että ilman paitaa.
  3. Sekä solmioon että paitaan pukeutuminen on liioittelua.
- Muuttujat **solmio** ja **paita** ovat binaarisia:
  voivat saada arvon 1 (tosi) tai 0 (epätosi)

Voidaanko kaikki rajoitteet *toteuttaa* yhtäaikaisesti?

Esimerkki: Luennoijan pukukoodiongelma

- Kolme rajoitetta jotka tulee toteuttaa:
  1. Selvästikään ei tule käyttää **solmio**ta ilman **paita**a.
  2. On epäkohteliasta olla sekä ilman **solmio**ta että ilman **paita**a.
  3. Sekä **solmio**on että **paita**an pukeutuminen on liioittelua.
- Muuttujat **solmio** ja **paita** ovat binaarisia:
  voivat saada arvon 1 (tosi) tai 0 (epätosi)

Voidaanko kaikki rajoitteet *toteuttaa* yhtäaikaisesti?

Esimerkki: Luennoijan pukukoodiongelma

- Kolme rajoitetta jotka tulee toteuttaa:
  1. Selvästikään ei tule käyttää **solmio**ta ilman **paita**a.
     (EI **solmio**) TAI **paita**
  2. On epäkohteliasta olla sekä ilman **solmio**ta että ilman **paita**a.
  3. Sekä **solmio**on että **paita**an pukeutuminen on liioittelua.
- Muuttujat **solmio** ja **paita** ovat binaarisia:
  voivat saada arvon 1 (tosi) tai 0 (epätosi)

Voidaanko kaikki rajoitteet *toteuttaa* yhtäaikaisesti?

Esimerkki: Luennoijan pukukoodiongelma

- Kolme rajoitetta jotka tulee toteuttaa:
  1. Selvästikään ei tule käyttää **solmio**ta ilman **paita**a.
     (EI **solmio**) TAI **paita**
  2. On epäkohteliasta olla sekä ilman **solmio**ta että ilman **paita**a.
     solmio TAI **paita**
  3. Sekä **solmio**on että **paita**an pukeutuminen on liioittelua.

- Muuttujat **solmio** ja **paita** ovat binaarisia:
  voivat saada arvon 1 (tosi) tai 0 (epätosi)

Voidaanko kaikki rajoitteet *toteuttaa* yhtäaikaisesti?

Esimerkki: Luennoijan pukukoodiongelma

- Kolme rajoitetta jotka tulee toteuttaa:
  1. Selvästikään ei tule käyttää **solmio**ta ilman **paita**a.
     (EI **solmio**) TAI **paita**
  2. On epäkohteliasta olla sekä ilman **solmio**ta että ilman **paita**a.
     solmio TAI **paita**
  3. Sekä **solmio**on että **paita**an pukeutuminen on liioittelua.
     (EI **solmio**) TAI (EI **paita**)
- Muuttujat **solmio** ja **paita** ovat binaarisia:
  voivat saada arvon 1 (tosi) tai 0 (epätosi)

Voidaanko kaikki rajoitteet *toteuttaa* yhtäaikaisesti?

Esimerkki: Luennoijan pukukoodiongelma

- Kolme rajoitetta jotka tulee toteuttaa:
  1. Selvästikään ei tule käyttää **solmio**ta ilman **paita**a.
     (EI **solmio**) TAI **paita**
  2. On epäkohteliasta olla sekä ilman **solmio**ta että ilman **paita**a.
     solmio TAI **paita**
  3. Sekä **solmio**on että **paita**an pukeutuminen on liioittelua.
     (EI **solmio**) TAI (EI **paita**)
- Muuttujat **solmio** ja **paita** ovat binaarisia:
  voivat saada arvon 1 (tosi) tai 0 (epätosi)

Voidaanko kaikki rajoitteet *toteuttaa* yhtäaikaisesti?

Luennoijan pukukoodi formaalimmin

1. (EI **solmio**) TAI **paita**
2. **solmio** TAI **paita**
3. (EI **solmio**) TAI (EI **paita**)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. (EI **solmio**) TAI **paita**
2. **solmio** TAI **paita**
3. (EI **solmio**) TAI (EI **paita**)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. (EI s) TAI **paita**
2. **solmio** TAI **paita**
3. (EI **solmio**) TAI (EI **paita**)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. (EI s) TAI **paita**
2. s TAI **paita**
3. (EI **solmio**) TAI (EI **paita**)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. (EI s) TAI **paita**
2. s TAI **paita**
3. (EI s) TAI (EI **paita**)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. (EI s) TAI p
2. s TAI p
3. (EI s) TAI (EI p)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. (EI s) TAI p
2. s TAI p
3. (EI s) TAI (EI p)

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. $\neg$ s TAI p
2. s TAI p
3. $\neg$ s TAI $\neg$ p

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. $\neg$ s TAI p
2. s TAI p
3. $\neg$ s TAI $\neg$ p

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. $\neg \; s \quad \vee \quad p$
2. $s \; \vee \; p$
3. $\neg \; s \quad \vee \quad \neg \; p$

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet:
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. $\neg\ s\ \vee\ p$
2. $s\ \vee\ p$
3. $\neg\ s\ \vee\ \neg\ p$

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet: $(\neg s \vee p)$, $(s \vee p)$, $(\neg s \vee \neg p)$
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. $\neg\ s\ \vee\ p$
2. $s\ \vee\ p$
3. $\neg\ s\ \vee\ \neg\ p$

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet: $(\neg s \vee p)$, $(s \vee p)$, $(\neg s \vee \neg p)$
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Luennoijan pukukoodi formaalimmin

1. $\neg\ s\ \vee\ p$
2. $s\ \vee\ p$
3. $\neg\ s\ \vee\ \neg\ p$

- Muuttujat s (**solmio**) ja p (**paita**)
- EI: negaatio $\neg$
- TAI: disjunktio $\vee$

- Rajoitteet: $(\neg s \vee p)$, $(s \vee p)$, $(\neg s \vee \neg p)$
- Asetetaan arvot $s = 0$ ja $p = 1$
  $\Rightarrow$ jokainen rajoitteista toteutuu
  $\Rightarrow$ *arvojakelu* $s = 0$, $p = 1$ on ongelman *ratkaisu*

Esimerkki *lauselogiikan toteutuvuusongelman instanssista ja sen ratkaisusta*

Lauselogiikan toteutuvuusongelma (SAT)

- Rajoitteet muotoa $(l_1 \vee \cdots \vee l_k)$, missä $k \geq 1$
  - Kukin l_i on joko jokin binaarimuuttuja x tai sen negaatio $\neg x$
  - Kukin muuttuja voi saada *totuus*arvon 1 (*tosi*) tai 0 (*epätosi*)
- Tulkinta (semantiikka):
  - $\neg x$ on tosi joss muuttuja x on epätosi
  - Rajoite $(l_1 \vee \cdots \vee l_k)$ toteutuu kun jokin l_i on tosi

Lauselogiikan toteutuvuusongelma hakuongelmana

Syöte: joukko rajoitteita, eli *SAT-instanssi*
Ratkaisu:

- Arvojakelu joka *toteuttaa* jokaisen annetuista rajoitteista (jos olemassa)
- muuten “ei”

Lauselogiikan toteutuvuusongelma (SAT)

- Rajoitteet muotoa $(l_1 \vee \cdots \vee l_k)$, missä $k \geq 1$
  - Kukin l_i on joko jokin binaarimuuttuja x tai sen negaatio $\neg x$
  - Kukin muuttuja voi saada *totuus*arvon 1 (*tosi*) tai 0 (*epätosi*)
- Tulkinta (semantiikka):
  - $\neg x$ on tosi joss muuttuja x on epätosi
  - Rajoite $(l_1 \vee \cdots \vee l_k)$ toteutuu kun jokin l_i on tosi

Lauselogiikan toteutuvuusongelma hakuongelmana

Syöte: joukko rajoitteita, eli *SAT-instanssi*
Ratkaisu:

- Arvojakelu joka *toteuttaa* jokaisen annetuista rajoitteista (jos olemassa)
- muuten "ei"

Lauselogiikan toteutuvuusongelma (SAT)

- Rajoitteet muotoa $(l_1 \vee \cdots \vee l_k)$, missä $k \geq 1$
  - Kukin l_i on joko jokin binaarimuuttuja x tai sen negaatio $\neg x$
  - Kukin muuttuja voi saada *totuus*arvon 1 (*tosi*) tai 0 (*epätosi*)
- Tulkinta (semantiikka):
  - $\neg x$ on tosi joss muuttuja x on epätosi
  - Rajoite $(l_1 \vee \cdots \vee l_k)$ toteutuu kun jokin l_i on tosi

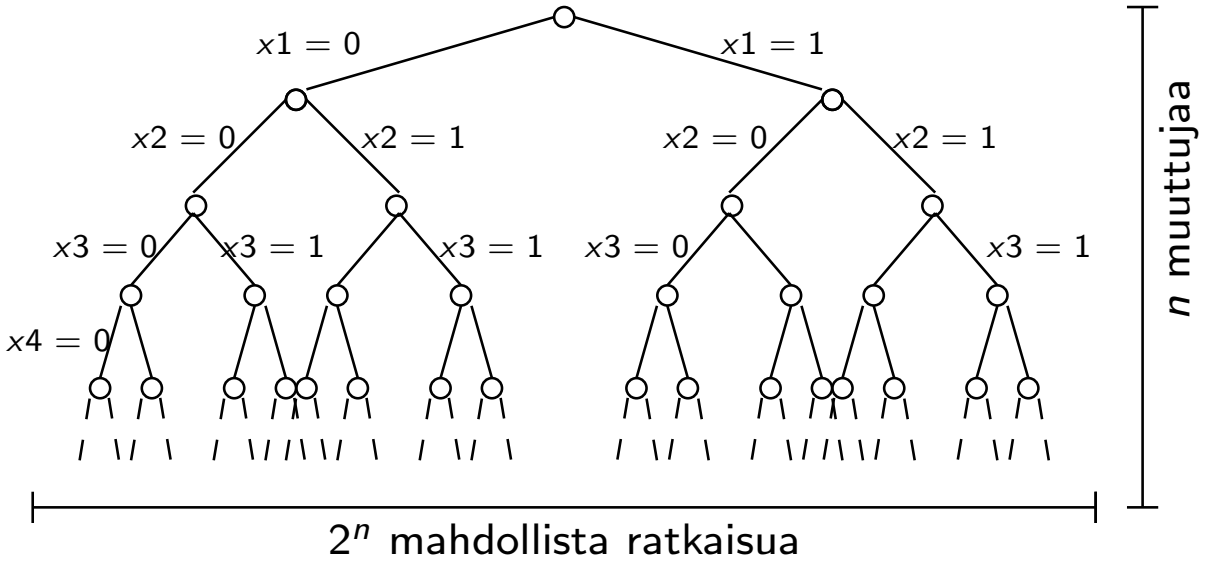
Lauselogiikan toteutuvuusongelma hakuongelmana

Syöte: joukko rajoitteita, eli *SAT-instanssi*
Ratkaisu:

- Arvojakelu joka *toteuttaa* jokaisen annetuista rajoitteista (jos olemassa)
- muuten “ei”

SAT ja laskennallinen vaativuus

- n muuttujaa, kukin voi saada arvon 1 tai 0
  $\Rightarrow 2^n$ mahdollista ratkaisuehdokasta

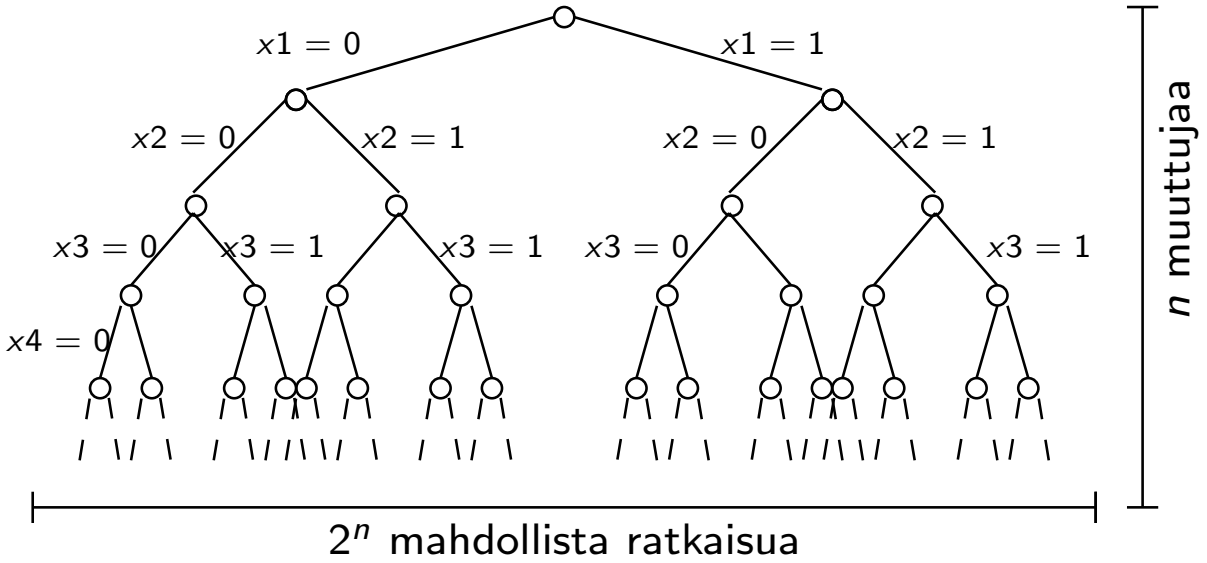


SAT yksi tunnetuimmista *NP-täydellisistä* ongelmista

- Polynomiaikaista ratkaisualgoritmia ei tunneta
- Kääntöpuolena: useita reaalimaailman ongelmia voidaan kuvata kompaktisti SAT-ongelmana

SAT ja laskennallinen vaativuus

- n muuttujaa, kukin voi saada arvon 1 tai 0
  $\Rightarrow 2^n$ mahdollista ratkaisuehdokasta

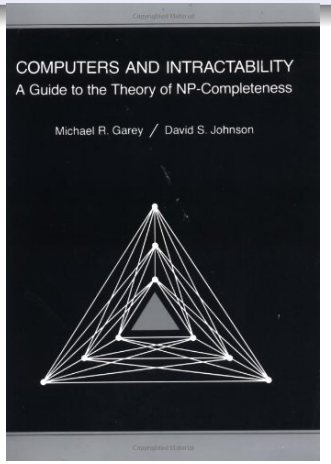


SAT yksi tunnetuimmista *NP-täydellisistä* ongelmista

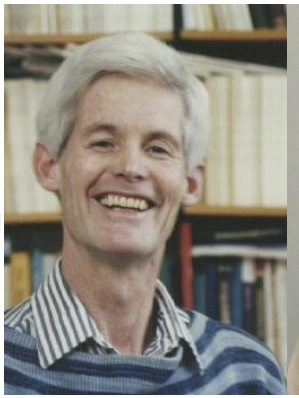
- Polynomiaikaista ratkaisualgoritmia ei tunneta
- Kääntöpuolena: useita reaalimaailman ongelmia voidaan kuvata kompaktisti SAT-ongelmana

SAT teoriassa

Keskeinen ongelma



- Cook-Levin -teoreema
  - NP-täydellisyys
  - Polynomiaikaiset reduktio
- Keskeinen P vs NP -kysymyksen suhteen
  - Yksi *Clay Instituten Millenium -ongelmista*
  - Ongelman tarkaisusta luvassa $ 1 miljoonan palkinto



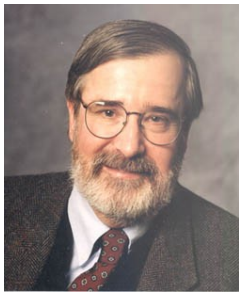
Toteutuvuustarkastus käytännössä

Yksi modernin tietojenkäsittelytieteen menestystarinoista

Satoja käytännön sovelluksia:

Hardware model checking, **Automated Planning**

Software model checking; Termination analysis of term-rewrite systems; Test pattern generation (testing of software & hardware); Model finding; Symbolic trajectory evaluation; Knowledge representation; Games (n-queens, sudoku, etc.); Haplotype inference; Pedigree checking; Equivalence checking; Delay computation; Fault diagnosis; Digital filter design; Noise analysis; Cryptanalysis; Inversion attacks on hash functions; Graph coloring; Traveling salesperson; van der Waerden numbers; itemset mining; etc. etc.



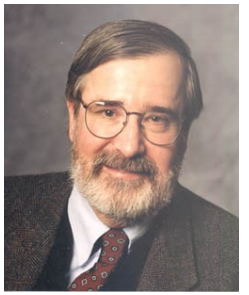
Toteutuvuustarkastus käytännössä

Yksi modernin tietojenkäsittelytieteen menestystarinoista

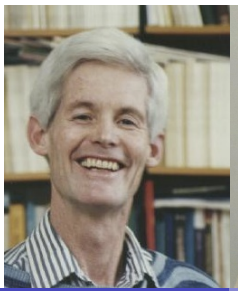
Satoja käytännön sovelluksia:

Hardware model checking, **Automated Planning**

Software model checking; Termination analysis of term-rewrite systems; Test pattern generation (testing of software & hardware); Model finding; Symbolic trajectory evaluation; Knowledge representation; Games (n-queens, sudoku, etc.); Haplotype inference; Pedigree checking; Equivalence checking; Delay computation; Fault diagnosis; Digital filter design; Noise analysis; Cryptanalysis; Inversion attacks on hash functions; Graph coloring; Traveling salesperson; van der Waerden numbers; itemset mining; etc. etc.



Polynomial-time reductions meet the real world!



SAT ja Teollisuudesta kumpuavat ongelmat

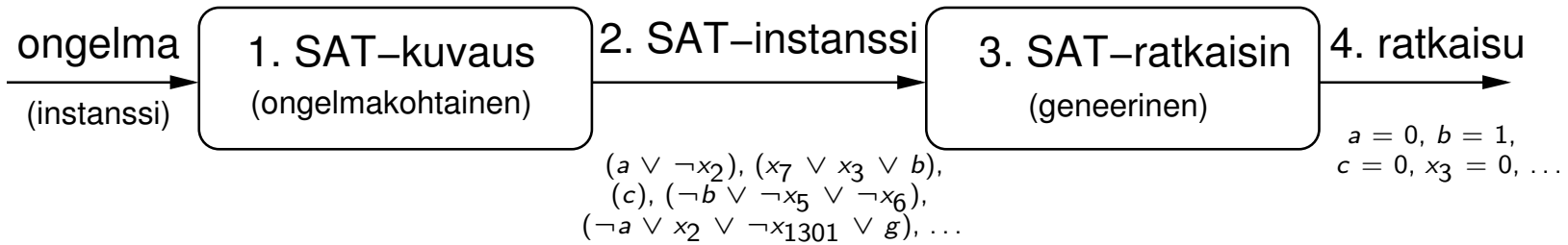
SAT-solverit keskeisessä osassa monenlaisissa teollisissa logiikkapiiri- ja ohjelmistoverifiointiongelmissa

- Epäsuorasti vaikuttamassa jokapäiviseen elämäämme!

Examples:

- **Intel core I7 processor design**ed with the help of SAT solvers
  [Kaivola et al, CAV 2009]
- **Windows 7 device drivers verified** using SAT related technology
  [De Moura and Bjorner, IJCAR 2010]
- **The Eclipse open platform** uses SAT technology for resolving dependencies between components
  [Le Berre and Rapicault, IWOCE 2009]

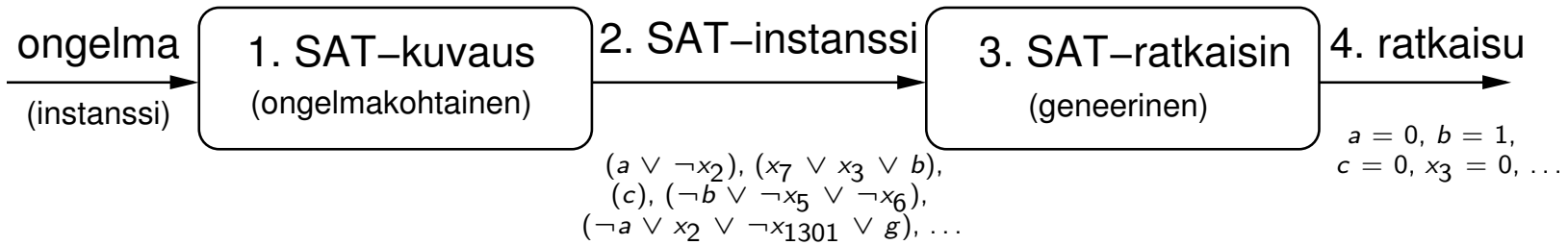
SAT-pohjainen ongelmanratkaisuvuo



1. **Kuvaus**: ratkaistava ongelma esitetään SAT-instanssina
   - *SAT-instanssin ratkaisujen tulee vastata alkuperäisen ongelman ratkaisuja*
2. SAT-instanssi syötetään *SAT-ratkaisimelle*
   - Algoritmi SAT-ongelmalle
3. **SAT-ratkaisin** joko
   (a) tulostaa SAT-instanssin ratkaisun, tai
   (b) todistaa että SAT-instanssilla ei ole ratkaisuja
4. Mahdollinen ratkaisu tulkitaan alkuperäisen ongelman ratkaisuksi

Ei tarvetta kehittää erityisalgoritmia jokaiselle ratkaistavalle ongelmalle

SAT-pohjainen ongelmanratkaisuvuo



1. **Kuvaus**: ratkaistava ongelma esitetään SAT-instanssina
   - *SAT-instanssin ratkaisujen tulee vastata alkuperäisen ongelman ratkaisuja*
2. SAT-instanssi syötetään *SAT-ratkaisimelle*
   - Algoritmi SAT-ongelmalle
3. **SAT-ratkaisin** joko
   (a) tulostaa SAT-instanssin ratkaisun, tai
   (b) todistaa että SAT-instanssilla ei ole ratkaisuja
4. Mahdollinen ratkaisu tulkitaan alkuperäisen ongelman ratkaisuksi

Ei tarvetta kehittää erityisalgoritmia jokaiselle ratkaistavalle ongelmalle

SAT-ratkaisimien tehokkuudesta

- *SAT-ratkaisimet* tekevät *SAT-pohjaisen lähestymistavan* erittäin kilpailukykyiseksi monille vaikeille laskennallisille ongelmille
- Huikea kehitys viimeisten 20 vuoden aikana

100 muuttujasta ja 200 rajoitteesta *jopa*
$>$10,000,000 muuttujaan ja 50,000,000 rajoitteeseen


Grasp'96


zChaff'01


Minisat'03


Lingeling'11

SAT-solvereista

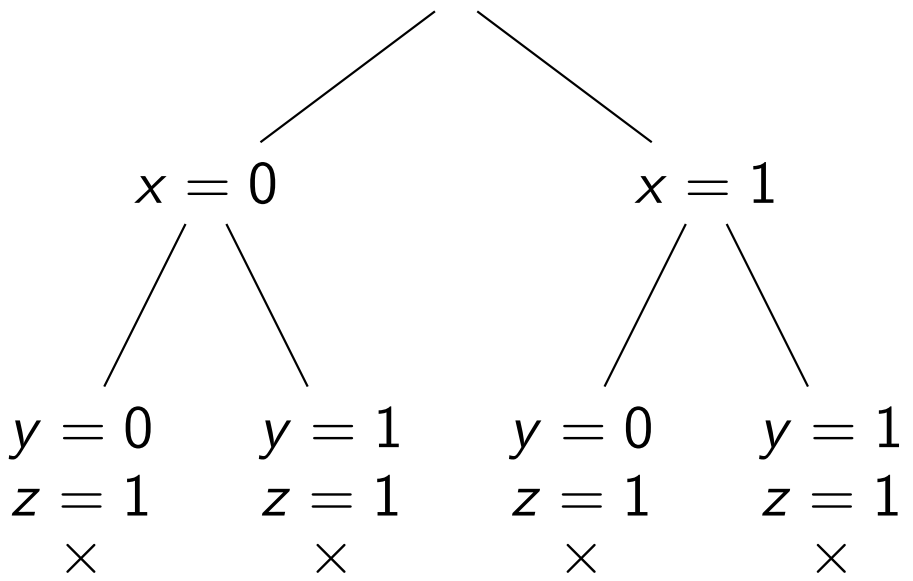
SAT-solverin käyttöesimerkki

Instance: aaai10-planning-ipc5-TPP-21-step11.cnf

```
c Lingeling SAT Solver
c Copyright (C) 2010-2014 Armin Biere JKU Linz Austria.
c read 99736 variables, 783991 clauses, 1708562 literals in 0.00 seconds
...
s UNSATISFIABLE
...
c 2.344 1% preprocessing 2%
c 140.829 30% inprocessing 98%
c ========================================
c 143.173 31% simplifying
c 320.824 69% search
c ========================================
c 464.001 100% all
c 4456232 decisions, 9603.9 decisions/sec
c 1181243 conflicts, 2545.8 conflicts/sec
c 2443039996 propagations, 5.3 megaprops/sec
c 464.0 seconds, 42.1 MB
```

DPLL: klassinen syvyyssuuntainen haku

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$

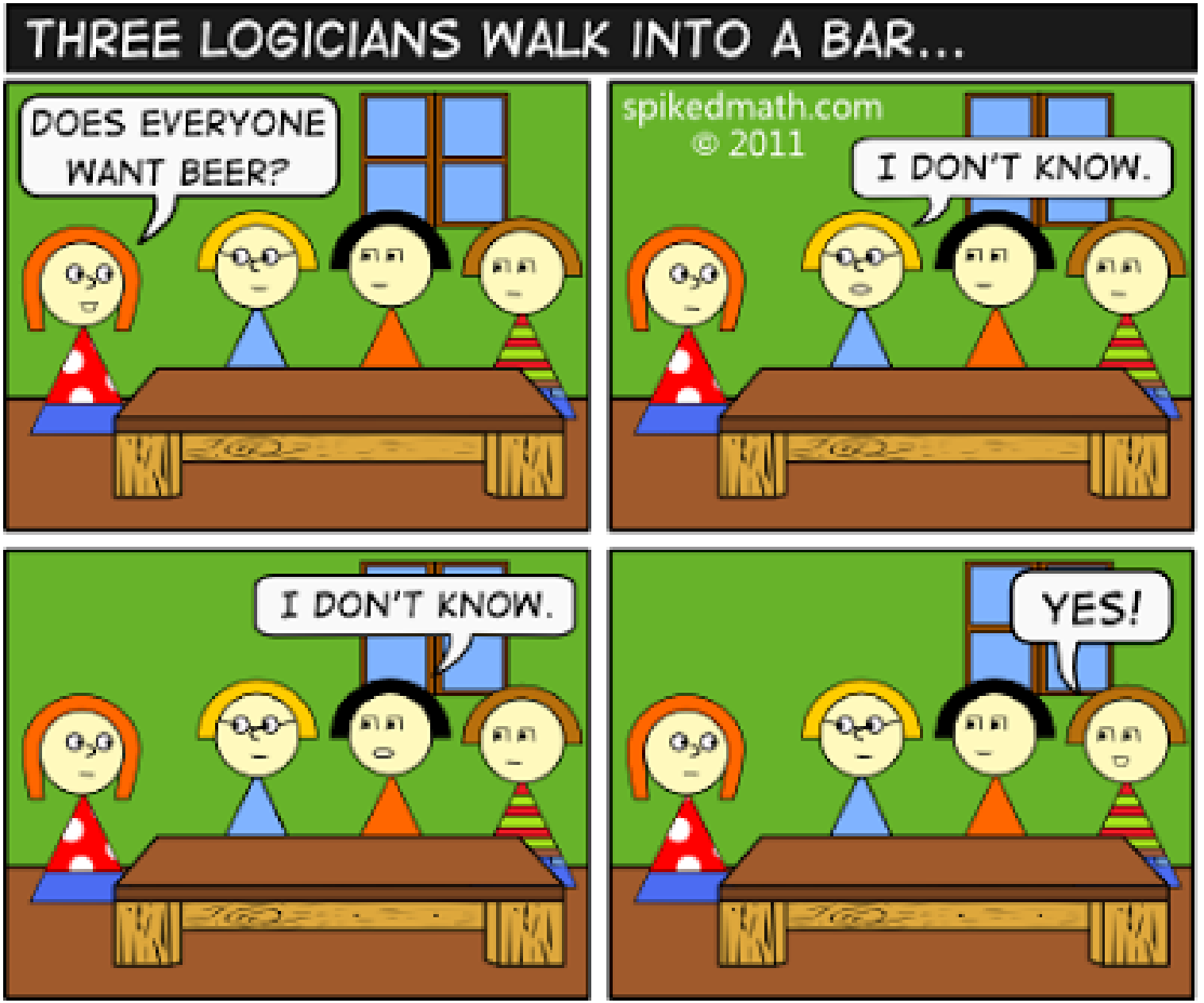


Esimerkissä:

Jokaisessa haarassa

- asetettava kahden muuttujan arvo *haarautumalla*
- kolmannen muuttujan arvo *päätellään yksikköpropagaatiolla*

Yksikköpropagaatio *unit propagation*

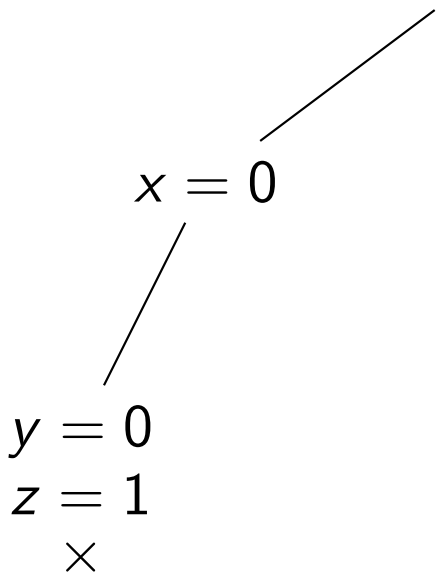


DPLL-esimerkki

Katsotaan vasemmanpuoleisinta haaraa

- Ensimmäinen klausuuli propagoi $z = 1$
- Toinen klausuuli propagoi ristiriidan $z = 0$

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$



Perääntyvä haku (*backtracking*)

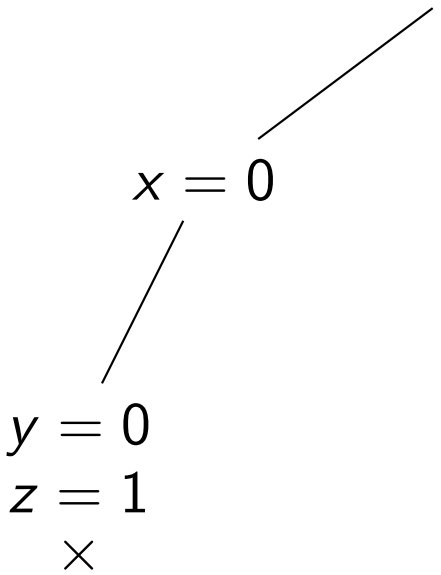
- Undo $y = 0$, $z = 1$
- Aseta $y = 1$ (toinen haara muuttujalle y)

Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$

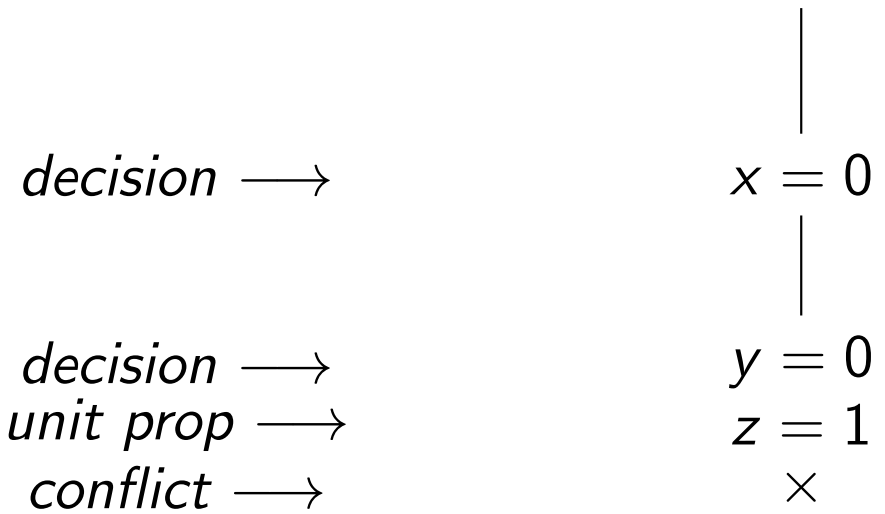


Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$

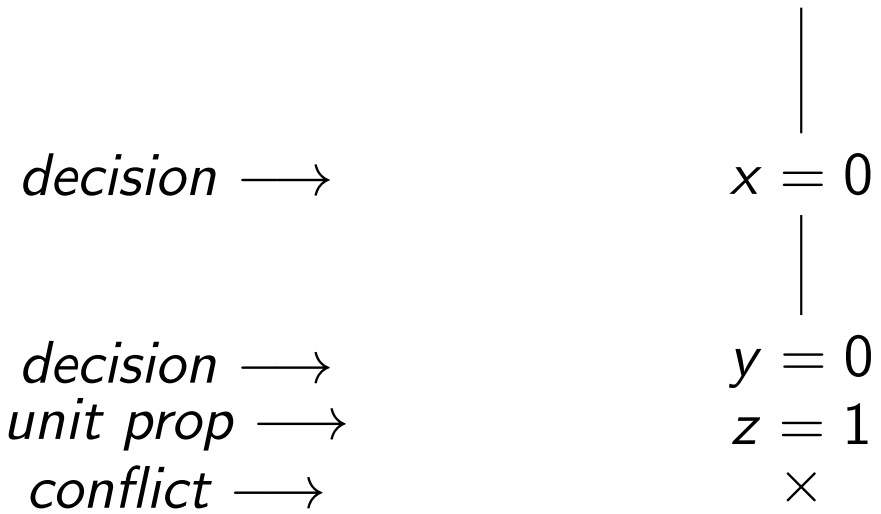


Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$



Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$

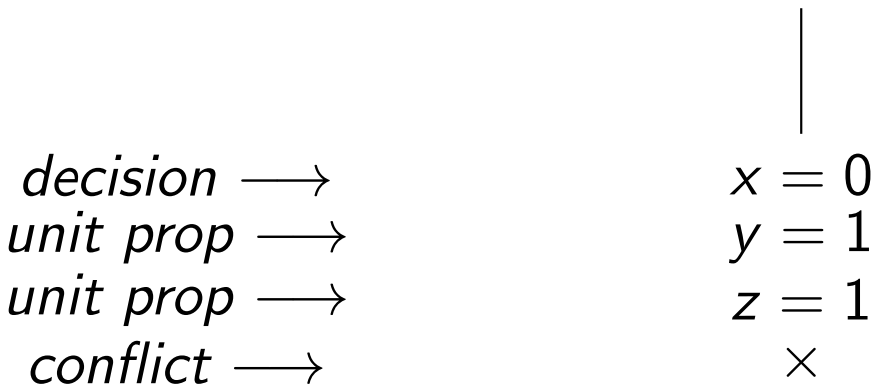
decision $\longrightarrow$

Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$

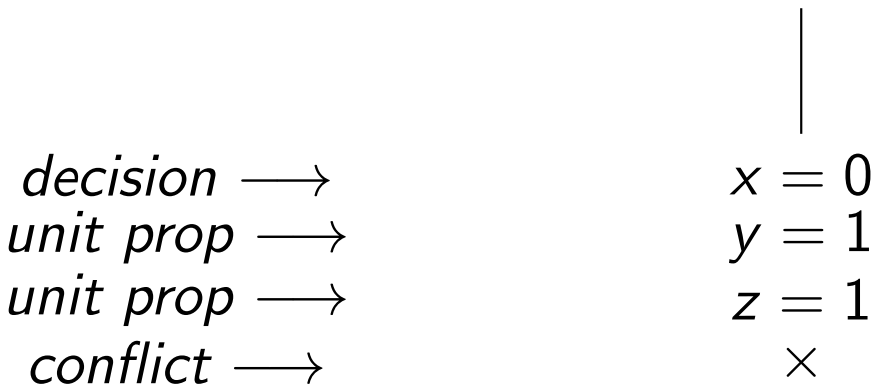


Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$, (x)



Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$, (x)

$$\textit{unit prop} \longrightarrow \qquad x = 1$$

Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$, (x)

unit prop $\longrightarrow$ $x = 1$

decision $\longrightarrow$ $z = 0$

Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$, (x) , (z)

$$\begin{array}{rc} \textit{unit prop} \longrightarrow & x = 1 \\ & \vert \\ \textit{decision} \longrightarrow & z = 0 \\ \textit{unit prop} \longrightarrow & y = 1 \\ \textit{conflict} \longrightarrow & \times \end{array}$$

Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$, (x) , (z)

$$\begin{aligned} \textit{unit prop} &\longrightarrow & x &= 1 \\ \textit{unit prop} &\longrightarrow & z &= 1 \\ \textit{unit prop} &\longrightarrow & y &= 1 \\ \textit{conflict} &\longrightarrow & \times & \end{aligned}$$

Conflict-Driven Clause Learning (CDCL)

CDCL: modernien SAT-solvereiden implementoima algoritmi

- *Oppiva hakumenetelmä*

$(x \vee y \vee z)$
$(x \vee y \vee \neg z)$
$(x \vee \neg y \vee z)$
$(x \vee \neg y \vee \neg z)$
$(\neg x \vee y \vee z)$
$(\neg x \vee y \vee \neg z)$
$(\neg x \vee \neg y \vee z)$
$(\neg x \vee \neg y \vee \neg z)$
learned:
$(x \vee y)$, (x) , (z) , $\emptyset$

$$\begin{array}{rl} \textit{unit prop} \longrightarrow & x = 1 \\ \textit{unit prop} \longrightarrow & z = 1 \\ \textit{unit prop} \longrightarrow & y = 1 \\ \textit{conflict} \longrightarrow & \times \end{array}$$

Termination:

Learned the empty clause

SAT Solver Competitions

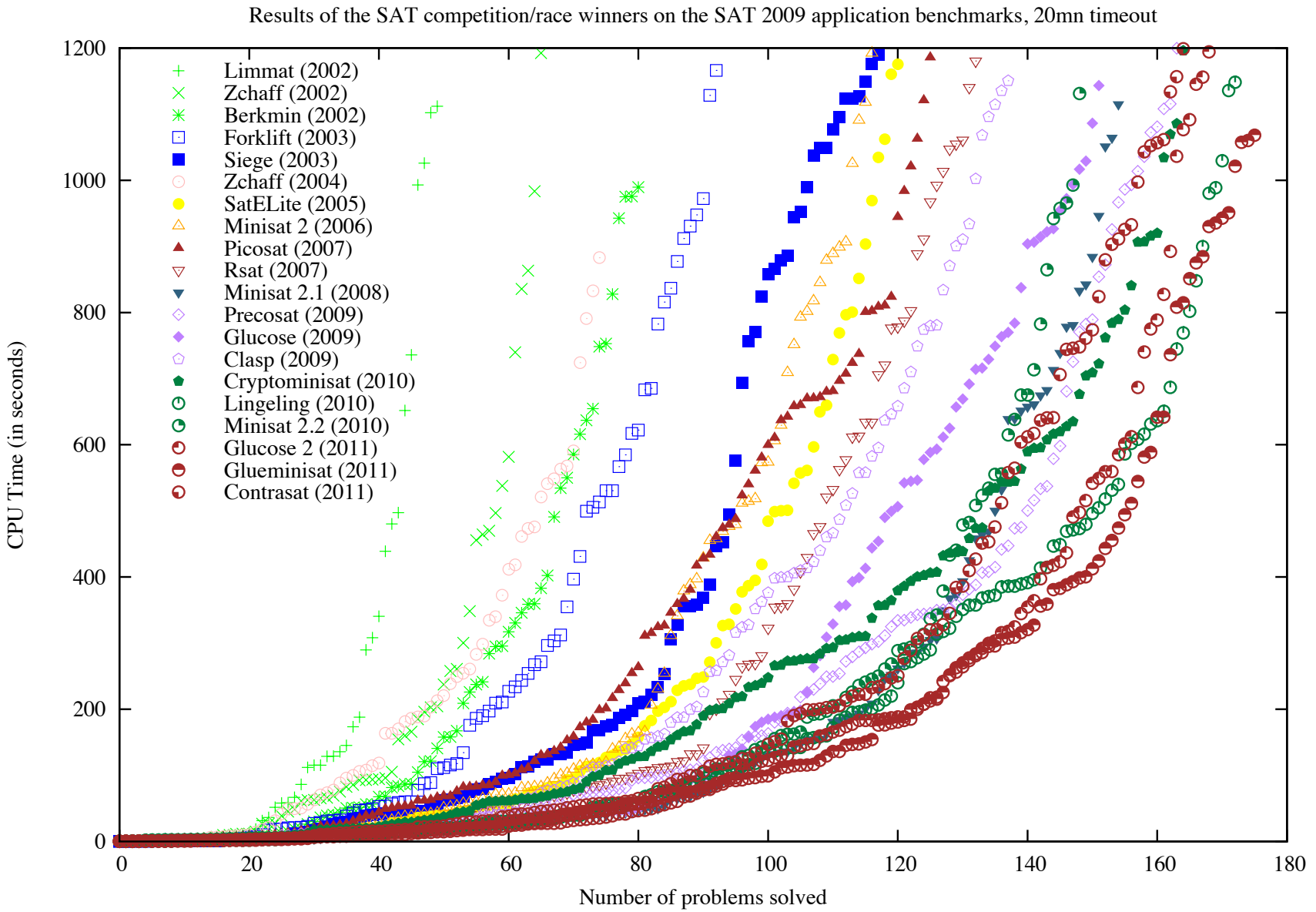
See `http://satcompetition.org/`

SAT 2014 competition

Organizing committee	Anton Belov, Daniel Diepold, Marijn Heule, Matti Järvisalo
Judges	Pete Manolios, Lakhdar Sais and Peter Stuckey
Proceedings	Descriptions of the solvers and benchmarks
Benchmarks	Application, Hard combinatorial, Random
Solvers	Source code available in EDACC

	Application			Hard combinatorial			Random		
	Gold	Silver	Bronze	Gold	Silver	Bronze	Gold	Silver	Bronze
Core solvers									
SAT+UNSAT	Lingeling	SWDiA5BY	Riss BlackBox	glueSplit_clasp	Lingeling	SparrowToRiss			
SAT	minisat_blbd	Riss BlackBox	SWDiA5BY	SparrowToRiss	CCAnr+glucose	SGSeq	Dimetheus	BalancedZ	CSCCSat2014
Certified UNSAT	Lingeling (druplig)	glucose	SWDiA5BY	Riss BlackBox	Lingeling (druplig)	glucose			
Core solvers, Parallel									
SAT+UNSAT	Plingeling	PeneLoPe	Treengeling	Treengeling	Plingeling	pmcSAT 2.0			
SAT							pprobSAT	Plingeling	CSCCSat2014
Minisat hack									
SAT+UNSAT	MiniSat_HACK_999ED	minisat_blbd	ROKKminisat						

SAT-solverien kehitys



Mallintaminen

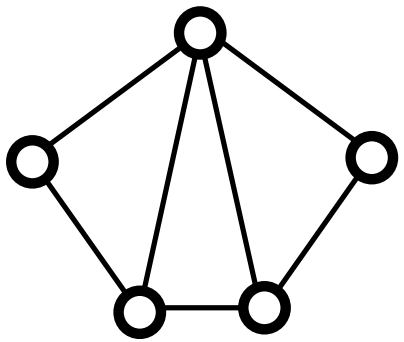
Verkon k-väritysongelma

k–väritysongelma *NP-täydellinen*

Syöte: Verkko $G = (V, E)$, kokonaisluku k (*värit* $1, \ldots, k$).
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

k-väritys:

- Jokaiselle solmulle $v \in V$: v väritetään täsmälleen yhdellä värillä.
- Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.



SAT-ratkaisimet käytännössä yksi tehokkaimpia tapoja värittää verkkoja

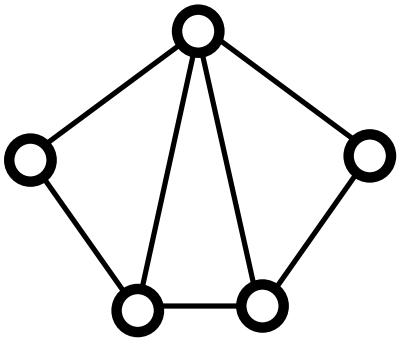
Verkon k-väritysongelma

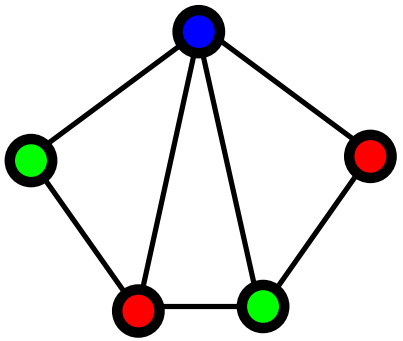
k–väritysongelma *NP-täydellinen*

Syöte: Verkko $G = (V, E)$, kokonaisluku k (*värit* $1, \ldots, k$).
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

k-väritys:

- Jokaiselle solmulle $v \in V$: v väritetään täsmälleen yhdellä värillä.
- Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.





3-väritys (1,2,3)

SAT-ratkaisimet käytännössä yksi tehokkaimpia tapoja värittää verkkoja

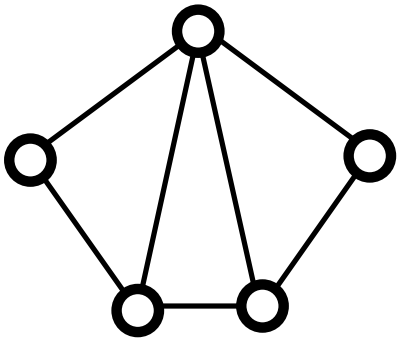
Verkon k-väritysongelma

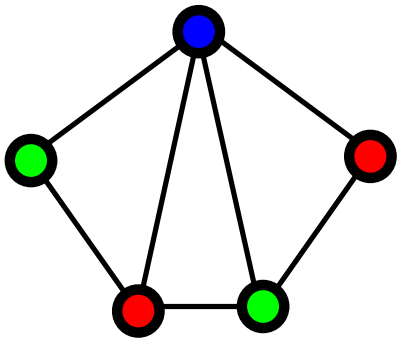
k–väritysongelma *NP-täydellinen*

Syöte: Verkko $G = (V, E)$, kokonaisluku k (*värit* $1, \ldots, k$).
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

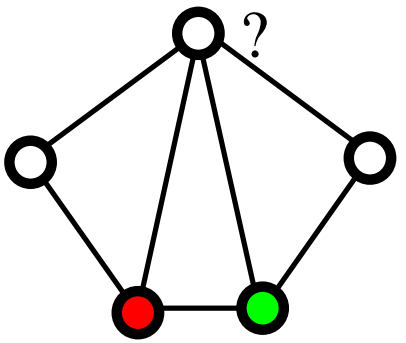
k-väritys:

- Jokaiselle solmulle $v \in V$: v väritetään täsmälleen yhdellä värillä.
- Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.





3-väritys (1,2,3)



ei 2-väritystä

SAT-ratkaisimet käytännössä yksi tehokkaimpia tapoja värittää verkkoja

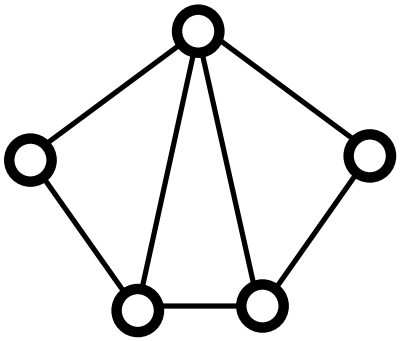
Verkon k-väritysongelma

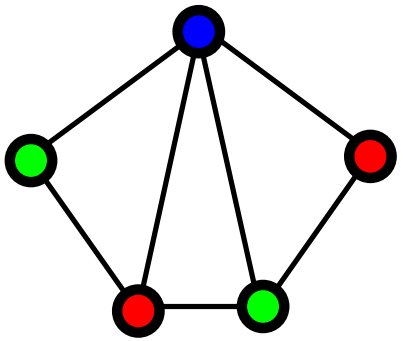
k–väritysongelma *NP-täydellinen*

Syöte: Verkko $G = (V, E)$, kokonaisluku k (*värit* $1, \ldots, k$).
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

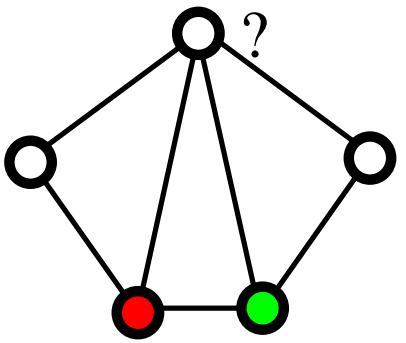
k-väritys:

- Jokaiselle solmulle $v \in V$: v väritetään täsmälleen yhdellä värillä.
- Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.





3-väritys (1,2,3)



ei 2-väritystä

SAT-ratkaisimet käytännössä yksi tehokkaimpia tapoja värittää verkkoja

3-värityksen kuvaus SAT-ongelmana

3-väritysongelma

Syöte: Verkko $G = (V, E)$.
Ratkaisu: G:n 3-väritys (jos olemassa), muuten "ei"

- Muuttujat: v_i = "solmu v väritetään värillä $i \in \{p, v, s\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä.
    – v väritetään enintään yhdellä värillä.
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

3-värityksen kuvaus SAT-ongelmana

3-väritysongelma

Syöte: Verkko $G = (V, E)$.
Ratkaisu: G:n 3-väritys (jos olemassa), muuten "ei"

3-väritysongelman SAT-kuvaus annetulle verkolla $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{p, v, s\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä.
    – v väritetään enintään yhdellä värillä.
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

3-värityksen kuvaus SAT-ongelmana

3-väritysongelma

Syöte: Verkko $G = (V, E)$.
Ratkaisu: G:n 3-väritys (jos olemassa), muuten "ei"

3-väritysongelman SAT-kuvaus annetulle verkolla $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{p, v, s\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä. $v_p \vee v_v \vee v_s$
    – v väritetään enintään yhdellä värillä.
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

3-värityksen kuvaus SAT-ongelmana

3-väritysongelma

Syöte: Verkko $G = (V, E)$.
Ratkaisu: G:n 3-väritys (jos olemassa), muuten "ei"

3-väritysongelman SAT-kuvaus annetulle verkolla $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{p, v, s\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä.
    – v väritetään enintään yhdellä värillä.

$$v_p \vee v_v \vee v_s$$
$$\neg v_p \vee \neg v_v$$
$$\neg v_p \vee \neg v_s$$
$$\neg v_v \vee \neg v_s$$

  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

3-värityksen kuvaus SAT-ongelmana

3-väritysongelma

Syöte: Verkko $G = (V, E)$.
Ratkaisu: G:n 3-väritys (jos olemassa), muuten "ei"

3-väritysongelman SAT-kuvaus annetulle verkolla $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{p, v, s\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä.
    – v väritetään enintään yhdellä värillä.

    $v_p \vee v_v \vee v_s$
    $\neg v_p \vee \neg v_v$
    $\neg v_p \vee \neg v_s$
    $\neg v_v \vee \neg v_s$
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

    $\neg v_p \vee \neg u_p$
    $\neg v_v \vee \neg u_v$
    $\neg v_s \vee \neg u_s$

k–värityksen kuvaus SAT-ongelmana

k–väritysongelma

Syöte: Verkko $G = (V, E)$, kokonaisluku k.
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

k-väritysongelman SAT-kuvaus annetulle verkolle $G = (V, E)$:

- Muuttujat: v_i = "solmu v väritetään värillä $i \in \{1, \ldots, k\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä.
    – v väritetään enintään yhdellä värillä.
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

k–värityksen kuvaus SAT-ongelmana

k–väritysongelma

Syöte: Verkko $G = (V, E)$, kokonaisluku k.
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

k-väritysongelman SAT-kuvaus annetulle verkolle $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{1, \ldots, k\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä. $v_1 \vee \cdots \vee v_k$
    – v väritetään enintään yhdellä värillä.
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

k–värityksen kuvaus SAT-ongelmana

k–väritysongelma

Syöte: Verkko $G = (V, E)$, kokonaisluku k.
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

k-väritysongelman SAT-kuvaus annetulle verkolle $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{1, \ldots, k\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä. $v_1 \vee \cdots \vee v_k$
    – v väritetään enintään yhdellä värillä. $\neg v_i \vee \neg v_j \ \forall i \neq j \in \{1, \ldots, k\}$
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.

k–värityksen kuvaus SAT-ongelmana

k–väritysongelma

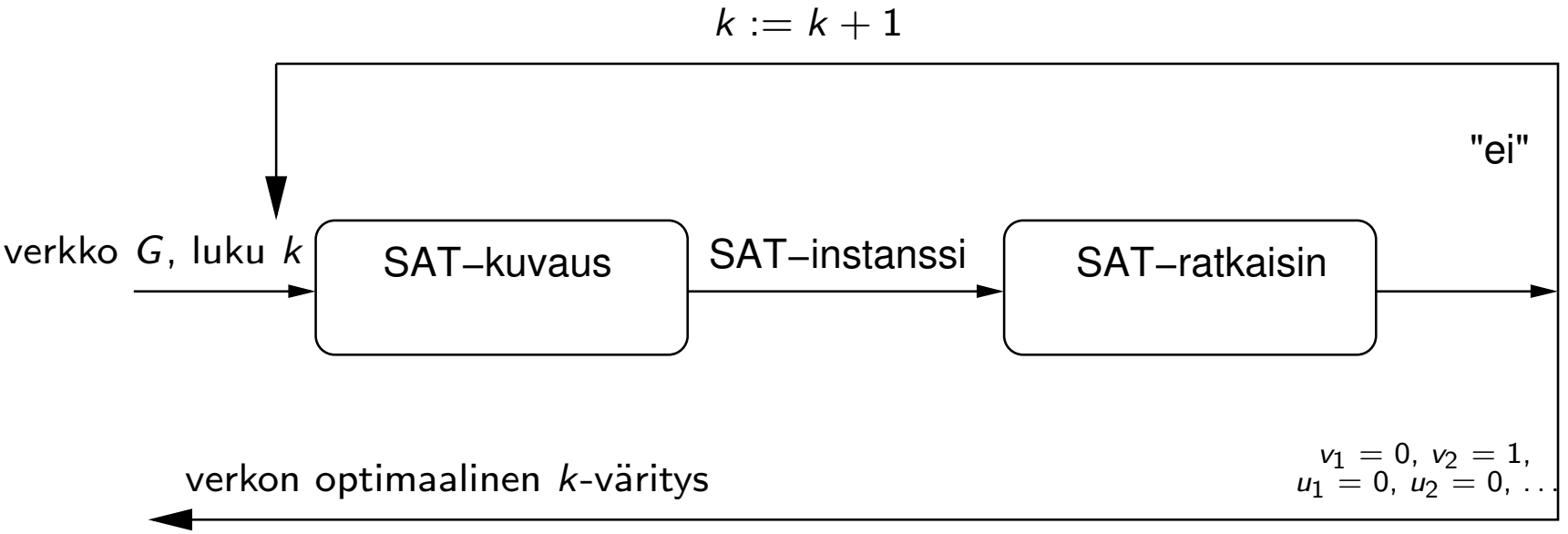
Syöte: Verkko $G = (V, E)$, kokonaisluku k.
Ratkaisu: G:n k-väritys (jos olemassa), muuten "ei"

k-väritysongelman SAT-kuvaus annetulle verkolle $G = (V, E)$:

- Muuttujat: $v_i =$ "solmu v väritetään värillä $i \in \{1, \ldots, k\}$"
- Rajoitteet:
  - Jokaiselle $v \in V$: v väritetään täsmälleen yhdellä värillä.
    Toisin sanoen:
    – v väritetään vähintään yhdellä värillä. $v_1 \vee \cdots \vee v_k$
    – v väritetään enintään yhdellä värillä. $\neg v_i \vee \neg v_j \ \forall i \neq j \in \{1, \ldots, k\}$
  - Jos kaari $(v, u) \in E$, niin v ja u väritetään eri väreillä.
    $\neg v_i \vee \neg u_i \ \forall i \in \{1, \ldots, k\}$

Optimointiongelman SAT-pohjainen ratkaiseminen: Verkon väritys mahdollisimman pienellä määrällä värejä

- **Syöte**: verkko G
- **Kysymys**: Pienin värien määrä k jolle G:lle on olemassa k–väritys?
- Alustetaan värien määrä $k := 1$

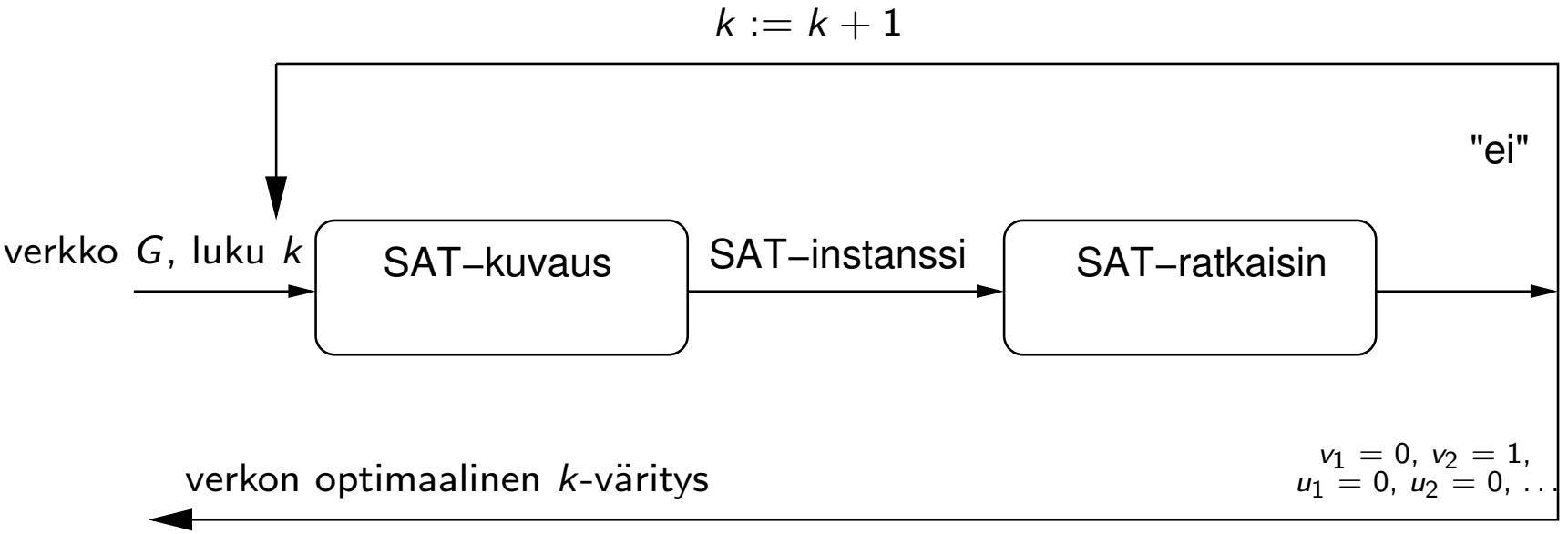


Lähestymistapa soveltuu moniin optimointiongelmiin

- Mallintarkastus:
  "Onko k:n mittaista suoritusta, joka vie ohjelman vikatilaan?"

Optimointiongelman SAT-pohjainen ratkaiseminen: Verkon väritys mahdollisimman pienellä määrällä värejä

- **Syöte**: verkko G
- **Kysymys**: Pienin värien määrä k jolle G:lle on olemassa k–väritys?
- Alustetaan värien määrä $k := 1$



Lähestymistapa soveltuu moniin optimointiongelmiin

- Mallintarkastus:
  "Onko k:n mittaista suoritusta, joka vie ohjelman vikatilaan?"

Mallintarkastuksesta automaattiseen suunnitteluun: Transitiojärjestelmien tarkastelu SAT-ongelmana

Hyvin yleinen ongelmanasettelu

Annettuna: yhden askeleen tilasiirtymäfunktio R, alkutila I, maalitila G.

Kysymys: Onko G saavutettavissa I:sta siirtymillä R?

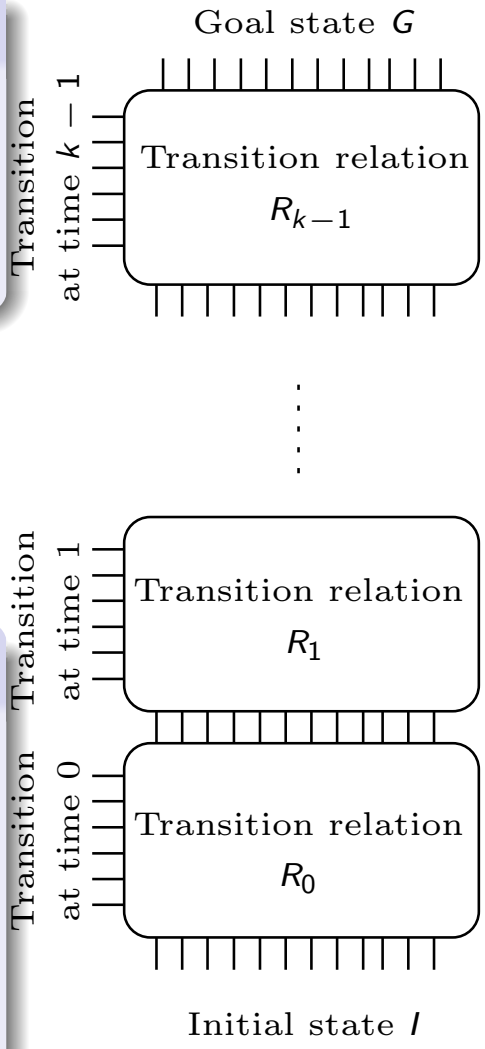
- Mallintarkastus: "*Paha* tila G saavutettavissa?"
- Suunnittelu: "Etsi suunnitelma päästä maalitilaan"

Mallintarkastuksesta automaattiseen suunnitteluun: Transitiojärjestelmien tarkastelu SAT-ongelmana

Hyvin yleinen ongelmanasettelu

Annettuna: yhden askeleen tilasiirtymäfunktio R, alkutila I, maalitila G.

Kysymys: Onko G saavutettavissa I:sta siirtymillä R?



- Mallintarkastus: "*Paha* tila G saavutettavissa?"
- Suunnittelu: "Etsi suunnitelma päästä maalitilaan"

SAT-pohjainen ratkaisu

$$\text{Onko} \quad I_0 \wedge \bigwedge_{i=0}^{k-1} R_i \wedge G_k \quad \text{toteutuva?}$$

- Ratkaisu kuvaa esim. virheellisen ohjelmasuorituksen!
- Ei ratkaisua $\rightsquigarrow$ oikeellisuus k askeleella

Yhteenveto

Yhteenveto

Moderni tekoälytutkimus

Yhteenveto

Moderni tekoälytutkimus

- **“Ei ainoastaan koneoppimista”**

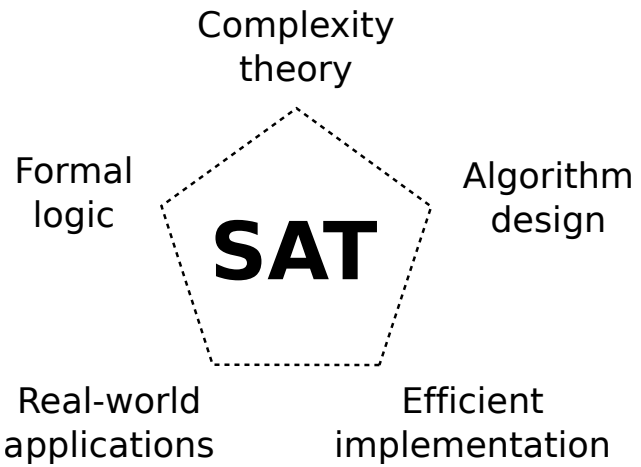
Yhteenveto

Moderni tekoälytutkimus

- **“Ei ainoastaan koneoppimista”**
- **Logiikka olennaissa osassa modernia tekoälytutkimus**

Lauselogiikan toteutuvuusongelma (SAT)

- Laskennallisesti haastava ongelma
- Useita laskennallisia ongelmia voidaan kuvata SAT-ongelmana
- SAT-solverit
  - Olennaisia työkaluja tekoäly- ja teollisten ongelmien ratkaisemisessa



Constraint Reasoning and Optimization @ TKTL

Constraint Reasoning and Optimization Group

`http://www.hiit.fi/cosco/coreo/`

Tutkimusteemoja:

- SAT-solvereiden ja muiden rajoiteratkaisumenetelmien kehitys
- *Teoreettisista tarkasteluista optimoituihin implementaatioihin*
- Vaikeiden, olennaisten laskennallisten ongelmien uudet ratkaisumenetelmät
  - Useilla moderneilla tekoälytutkimuksen alueilla:
    machine learning and data analysis, argumentation, decision theory, computational social choice, ...