Registrite allokeerimine

Registrite allokeerimine

Ülevaade

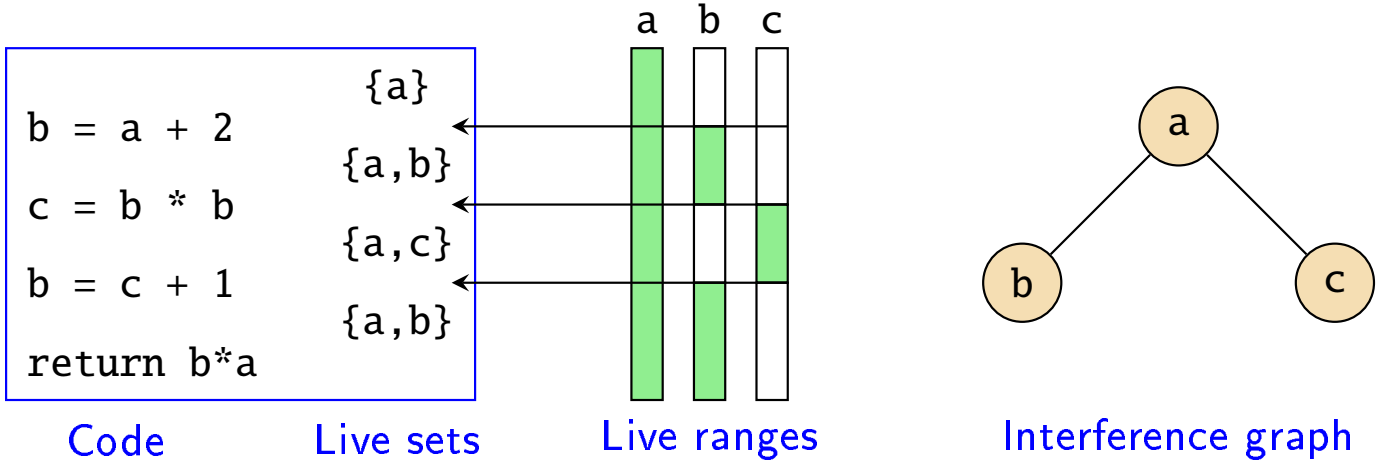
- Muutujate väärtusi võib hoida põhimälus või registrites.
  - Mällu kirjutamine ja mälust lugemine on registritega võrreldes väga aeglane (umbes kaks suurusjärku).
  - Registrite arv on rangelt piiratud (mõni kuni mõnikümmend).
- Registrite allokeerimise eesmärk on vähendada mälupöördumiste arvu hoides võimalikult paljude muutujate väärtusi registrites.
  - Otsustab milliseid väärtusi hoida registrites ja milliseid mälus.
  - Määrab registrites hoitavatele väärtustele konkreetsed registrid.

Registrite allokeerimine

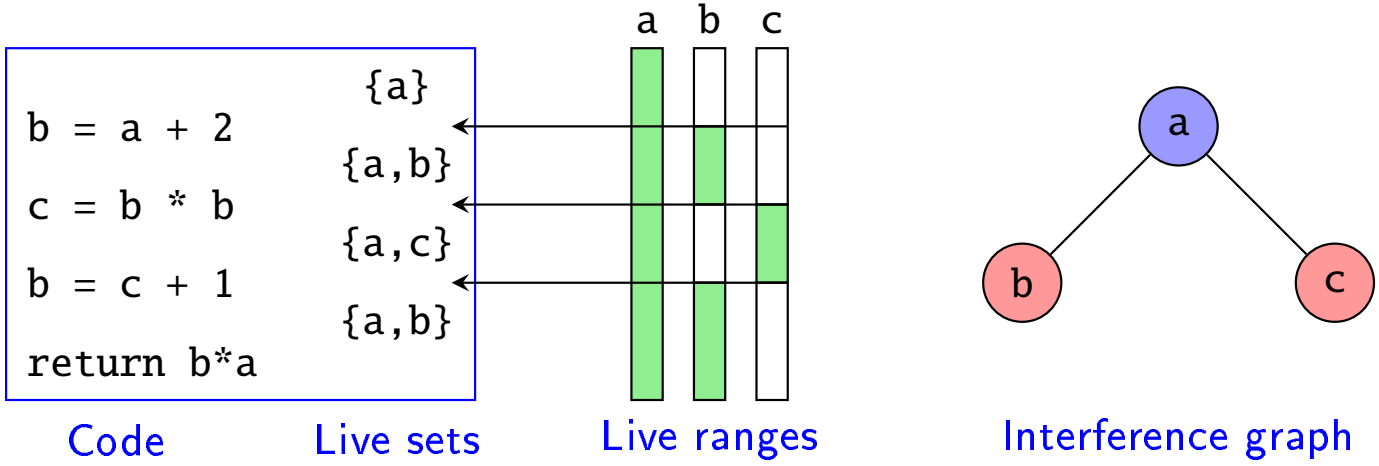
Põhitähelepanekud

- Tavaliselt on registreid vähem kui muutujaid.
- Samaaegselt elusolevad muutujad ei saa paikneda samas registris (muutujad interfereeruvad).
- Küll aga võivad muutujad mille eluajad ei kattu paikneda samas registris.
- Vastavaid kitsendusi saab esitada graafina:
  - tipud = muutujad;
  - kaared = samaaegselt elus olevate muutujate vahel;
  - nn. interferentsigraaf.
- Registrite määramine on selle graafi k värviga värvimise probleem (Lavrov 1962, Chaitin 1981)
  - k = registrite arv.

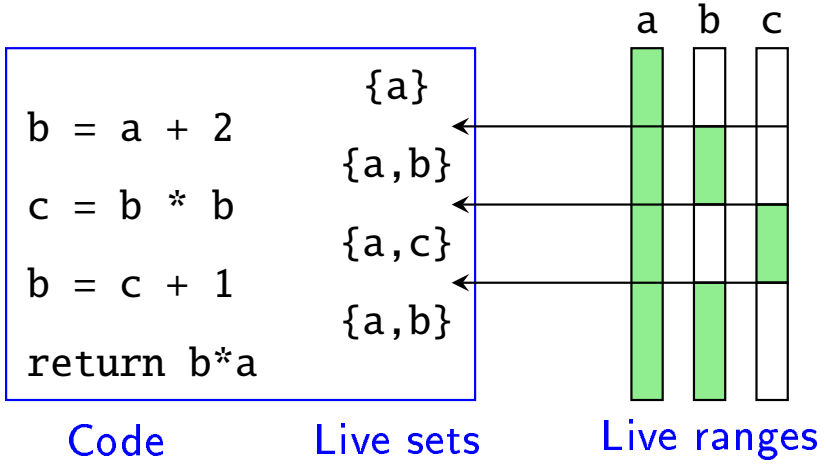
Registrite allokeerimine

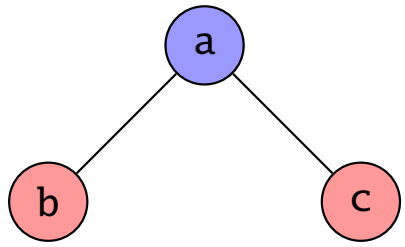


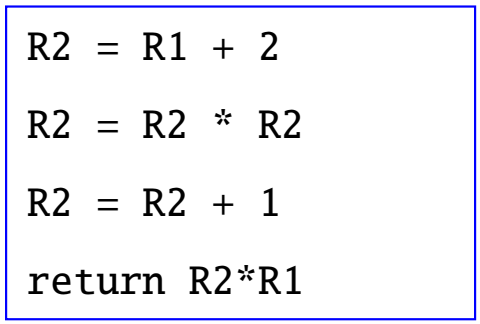
Registrite allokeerimine



Registrite allokeerimine





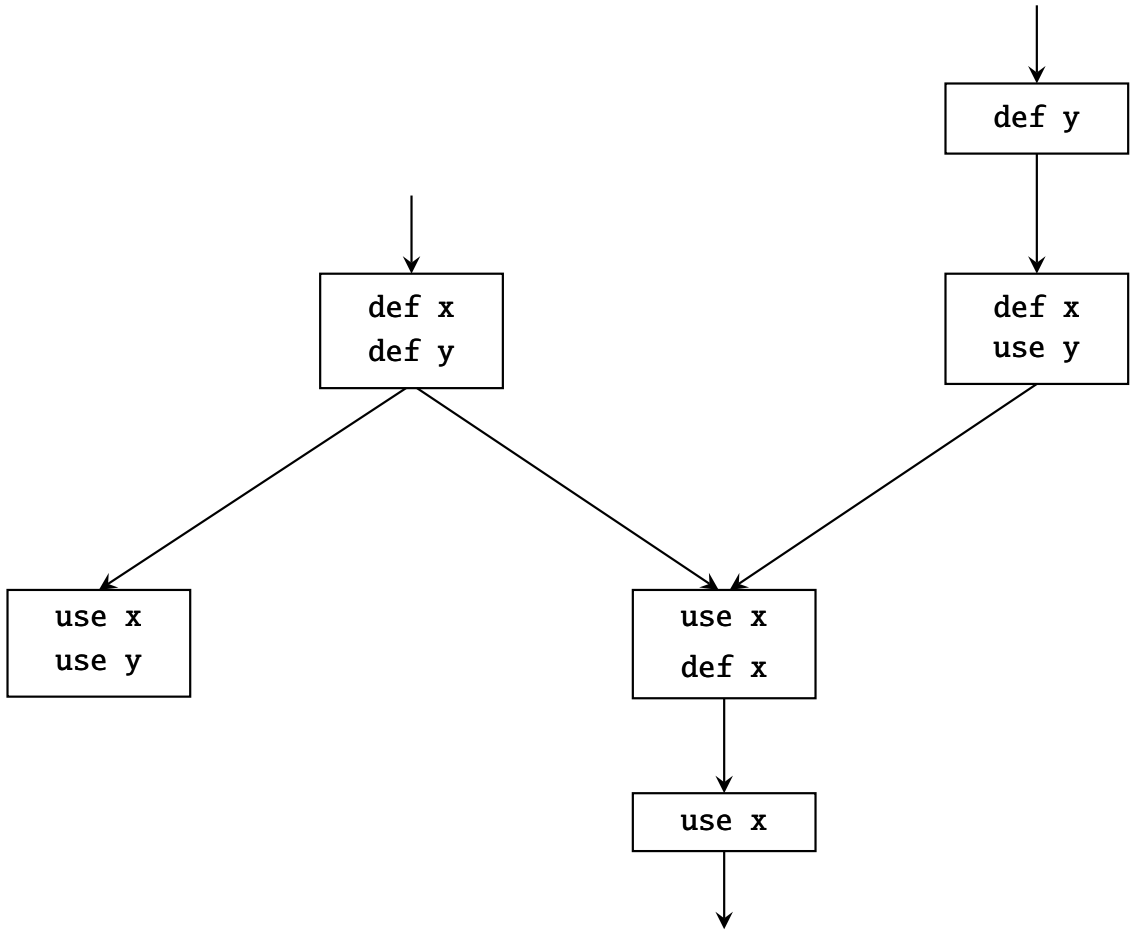


Registrite allokeerimine

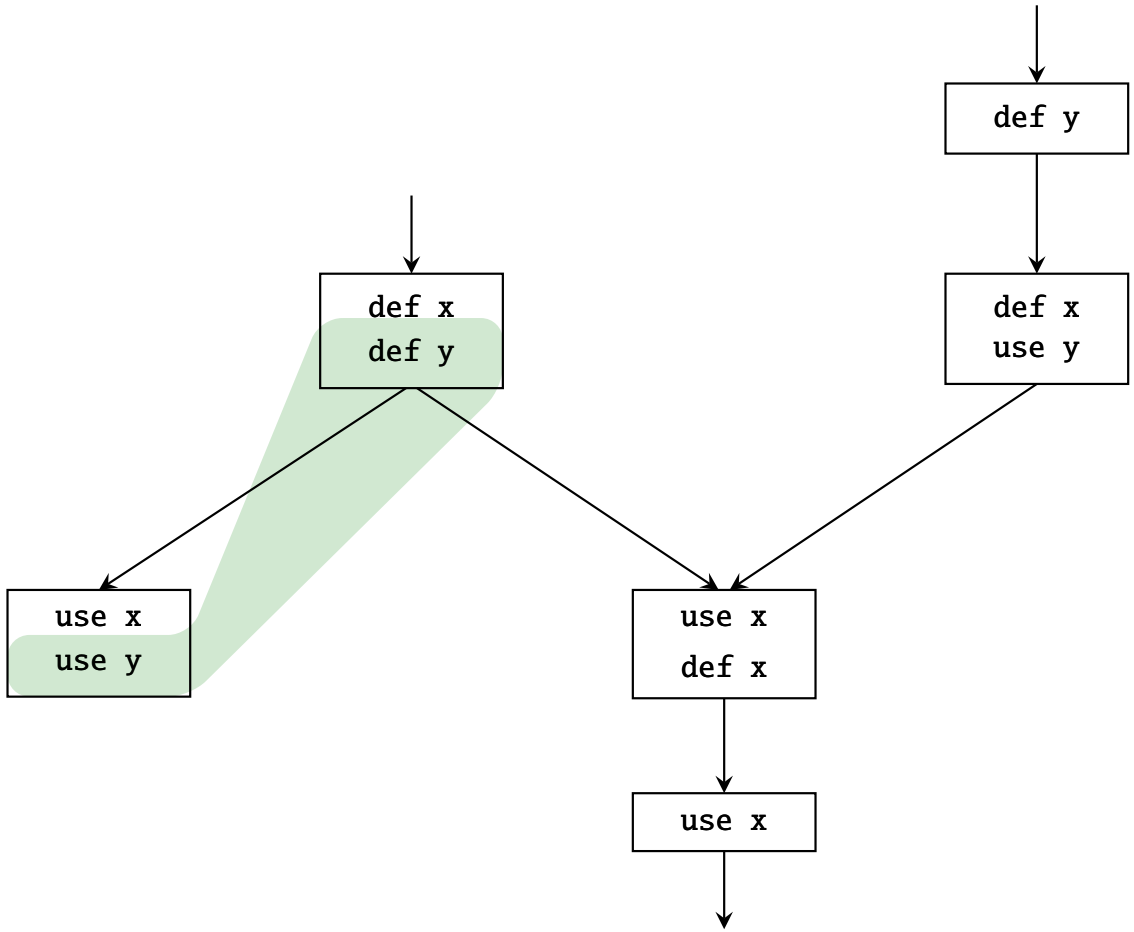
Graafi konstrueerimine

- Interferentsigraafi konstrueerimiseks tuleb leida muutujate elususpiirkonad.
- Baasploki piires registrite allokeerimisel on elususpiirkonnad lineaarsed.
  - ✔ Nii elususpiirkondade leidmine kui kattumise kontrollimine väga lihtne.
  - ✘ Baasploki alguses tuleb muutujad registritesse lugeda ning lõpus mällu salvestada.
- Globaalsel registrite allokeerimisel moodustavad elususpiirkonnad võrgu (web).
  - ✘ Elususpiirkondade leidmine keerulisem.
  - ✔ Võimaldab efektiivsemat registrite kasutust.

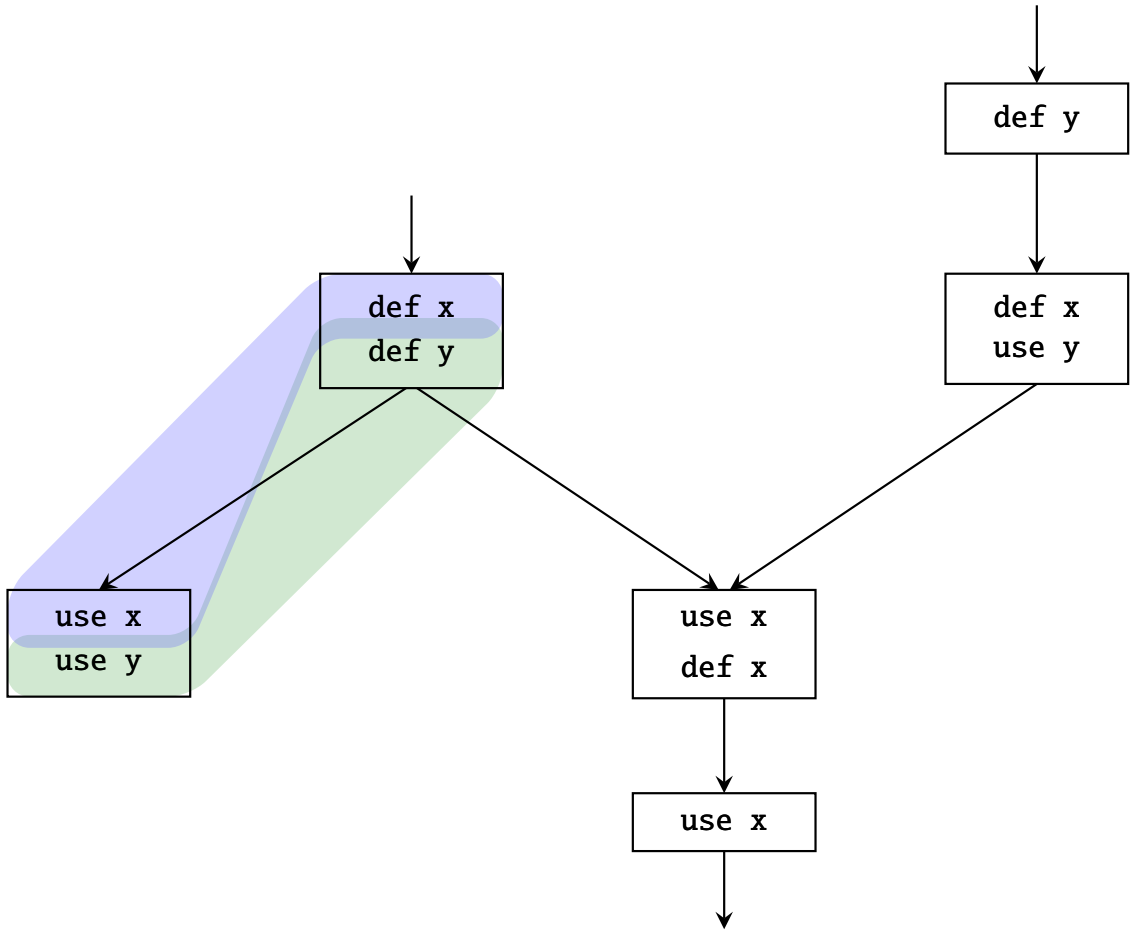
Registrite allokeerimine



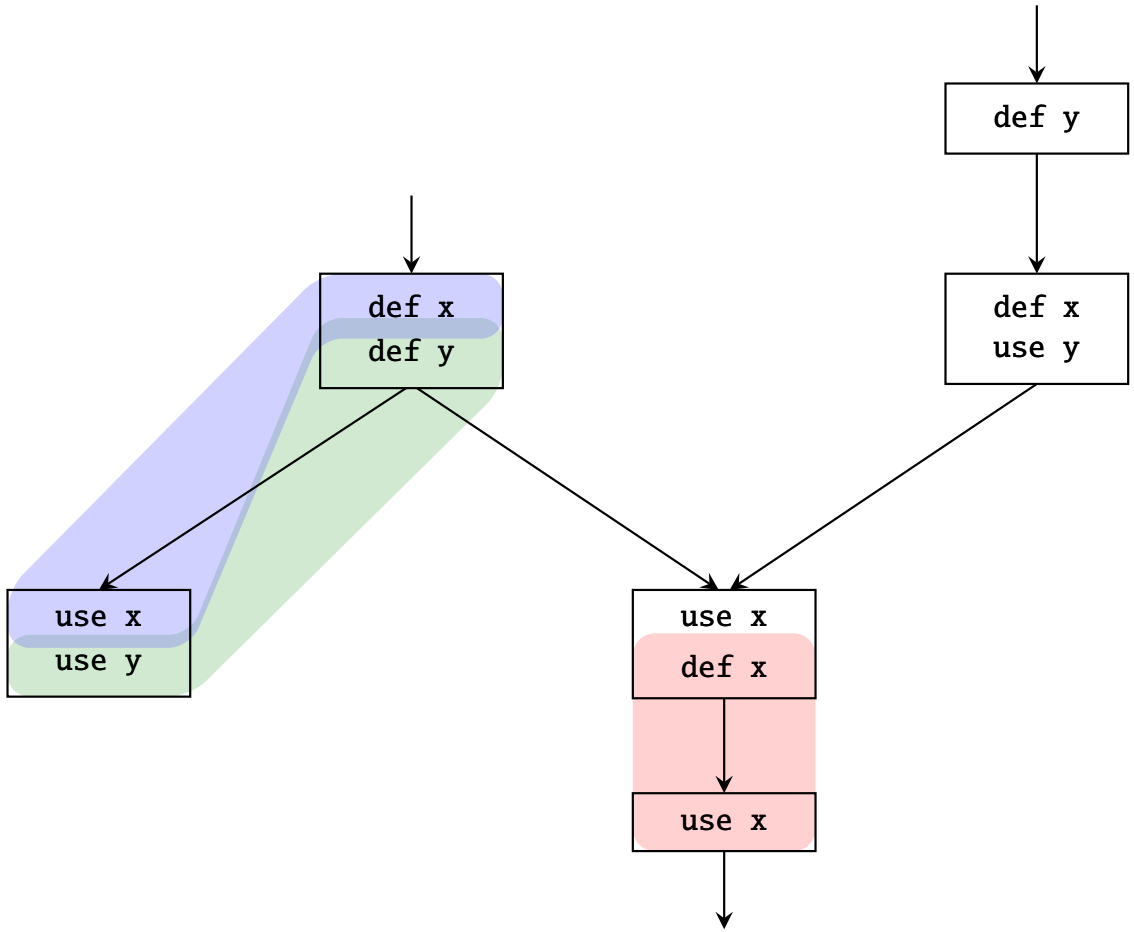
Registrite allokeerimine



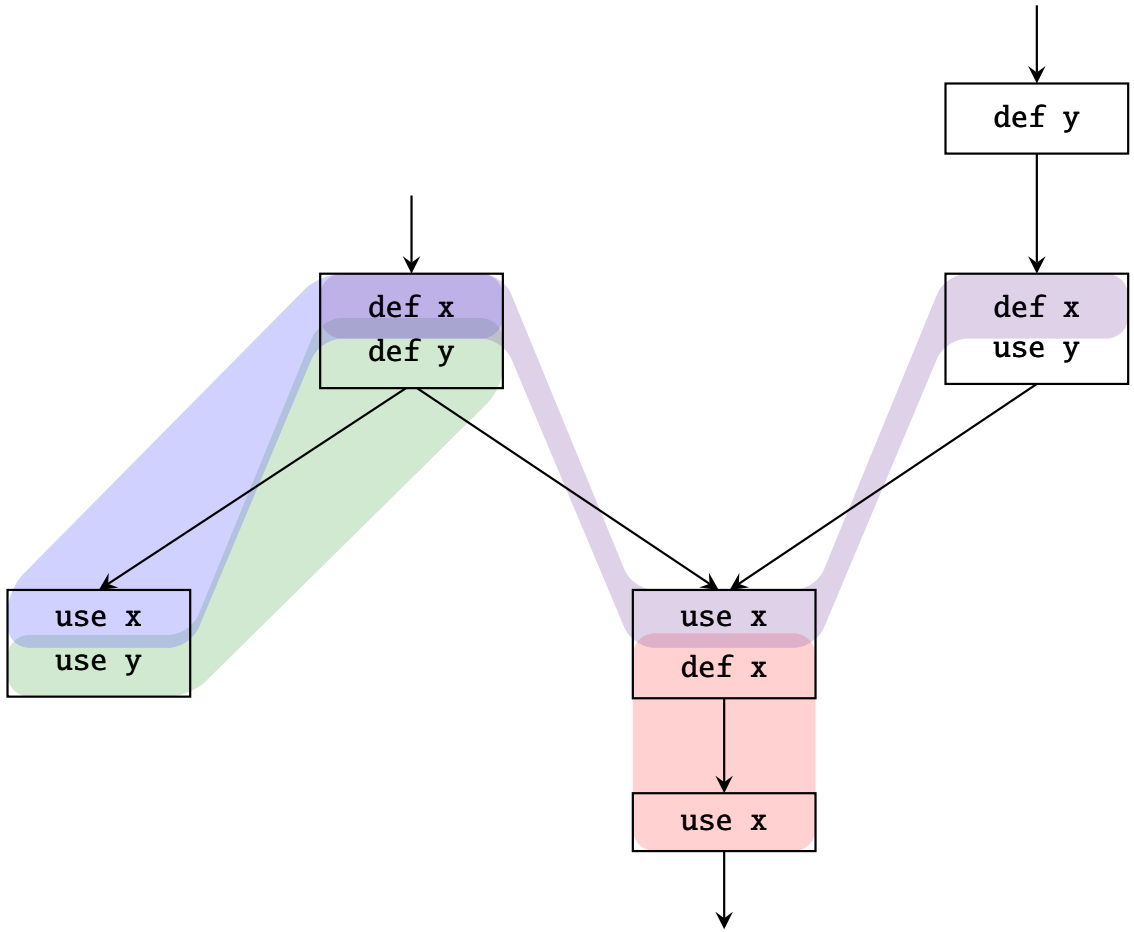
Registrite allokeerimine



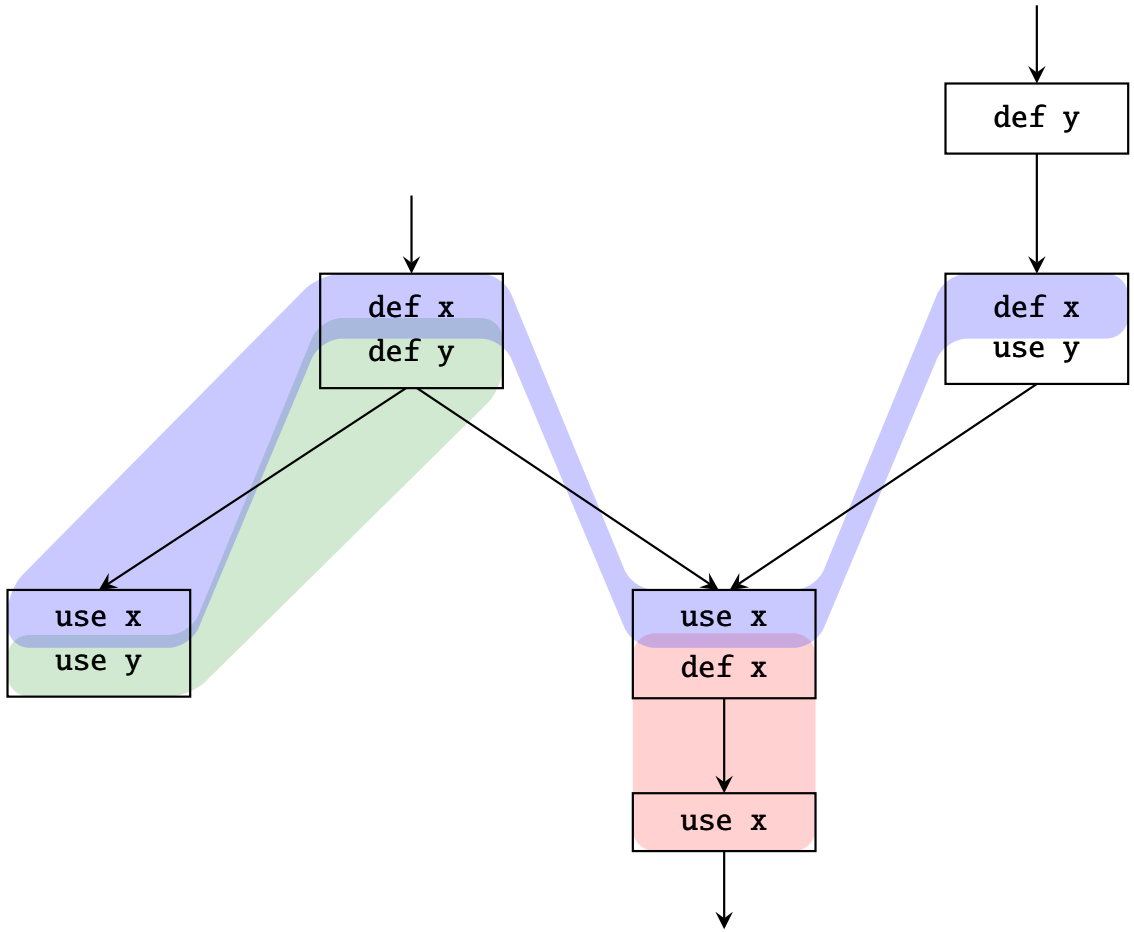
Registrite allokeerimine



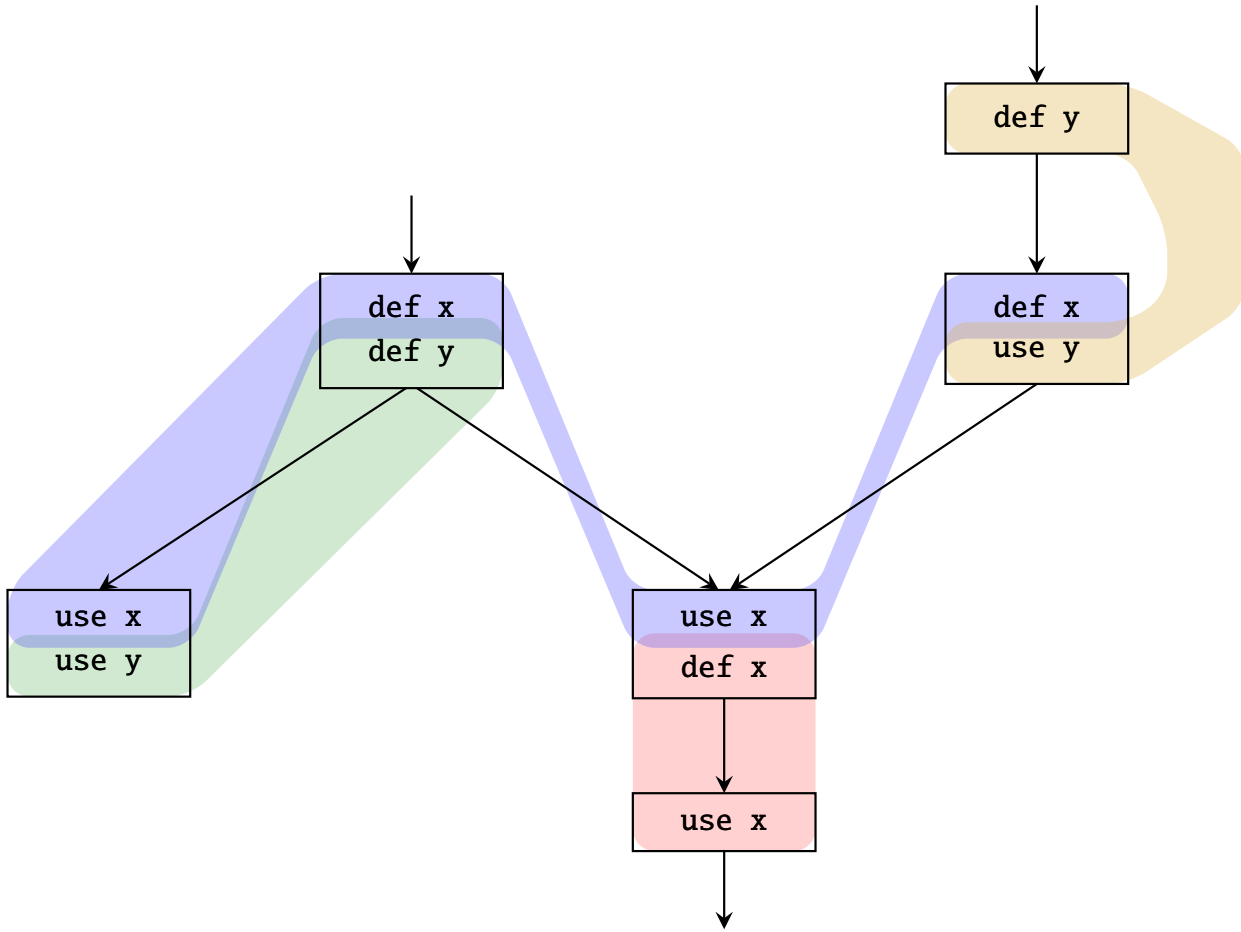
Registrite allokeerimine



Registrite allokeerimine

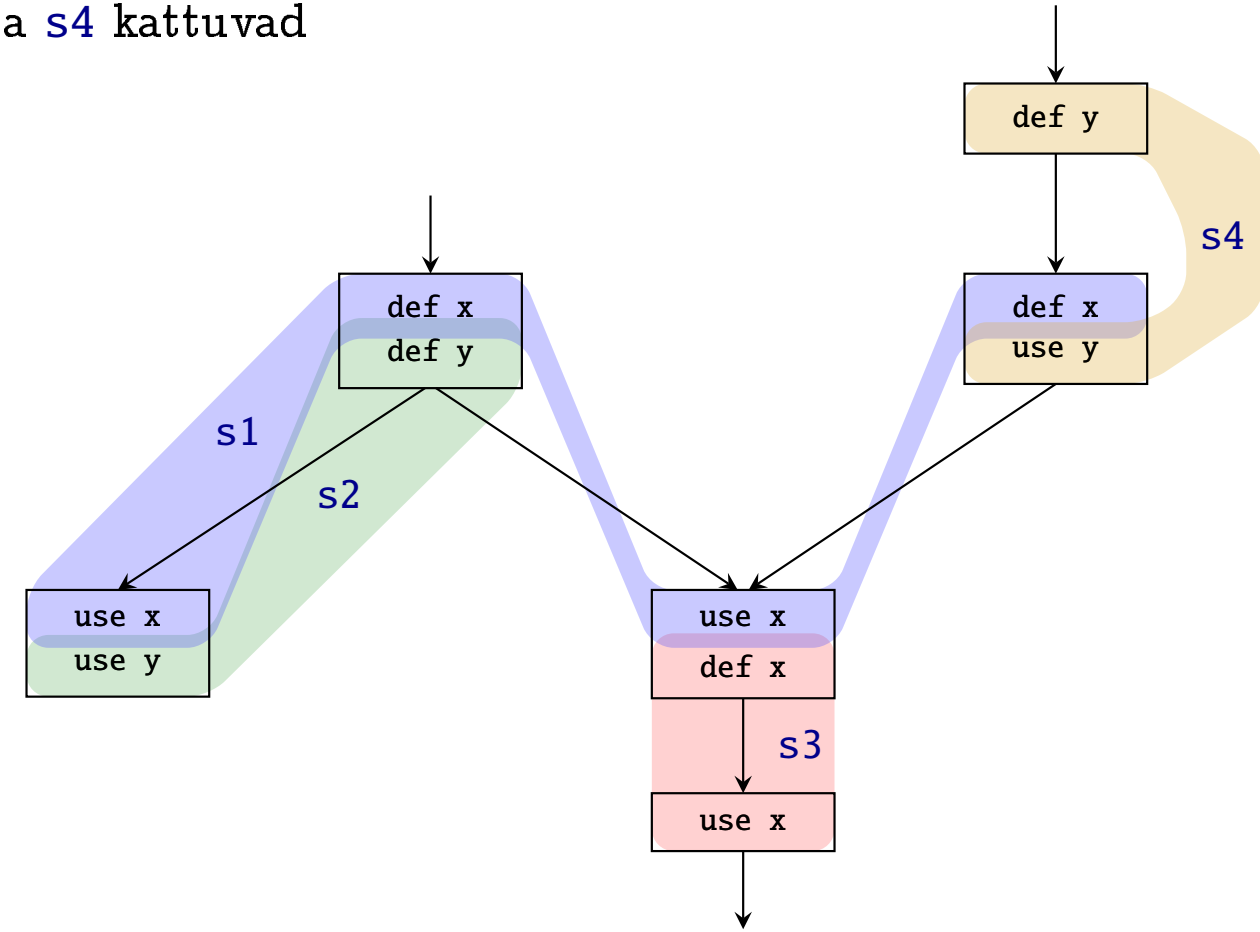


Registrite allokeerimine



Registrite allokeerimine

Võrgud s1 ja s2 kattuvad
Võrgud s1 ja s4 kattuvad



Graafi värvimine

Definitsioonid

- Graafi värvimiseks nimetatakse tema tippude märgendamist arvudega selliselt, et ükski tipp ei omaks ühegi oma naabriga sama märgendit.
- Graafi mis on värvitav k värviga nimetatakse k-aluseliseks.

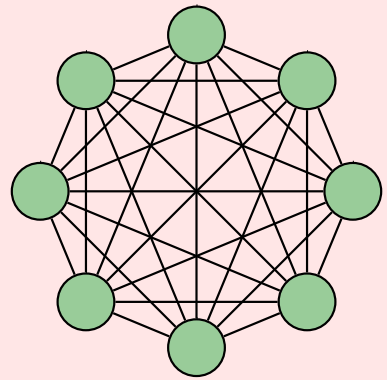
Põhiküsimused

- Kuidas efektiivselt leida graafi k-värvimine?
- Kas ja kuidas leida optimaalset (so. minimaalse värvide arvuga) värvimist?
- Mida teha, kui graafi värvimiseks pole piisavalt värve (so. registreid)?

Graafi värvimine

Probleem

Graafi värvimise ülesanne on NP-täielik.



Tähelepanekud

- Optimaalne algoritm töötab kõigil graafidel.
  - Halvima juhtumi graafi praktikas ei esine.
- Leiab alati minimaalse värvimise.
  - Tihti piisab ligikaudsest heuristilisest värvimisest.

Graafi värvimine

Probleem

Mida teha kui graaf pole k-aluseline?

- So. pole piisaval arvul registreid.
- Juhtub väga sageli.

Pillamine

- Valime muutuja ning hoiame teda registri asemel põhimälus (so. magasinis).
  - Seda protsessi nimetatakse pillamiseks (spilling).
- Muutuja kasutuskohtadesse genereerime ekstra koodi tema mälust ajutisse registrisse lugemiseks ja mällu kirjutamiseks.

Graafi värvimine

Idee

- Valime graafis tipu mille järk on väiksem kui k.
  - See tipp on k-värvitav!
- Eemaldame selle tipu (ja kõik temaga seotud servad) graafist.
  - Kõigi naabertippude järk väheneb ühe võrra.
  - Võib tekkida juurde uusi tippe mille järk on k-st väiksem.
- Kui kõik tipud on vähemalt k naabriga, siis valime tipu mille pillame mällu ja jätkame.

Graafi värvimine

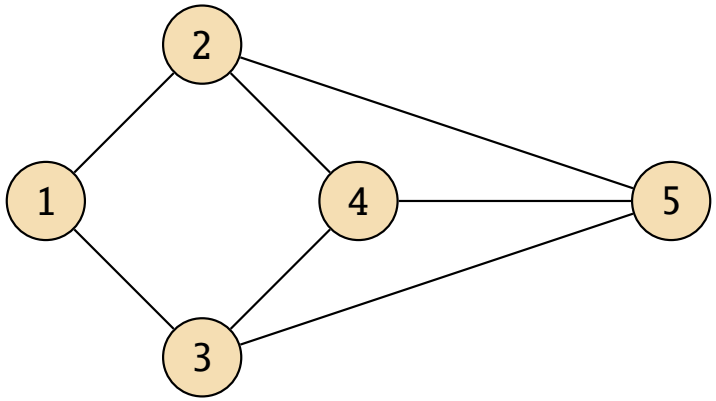
Chaitini algoritm

1. Kuni leidub tippe mille järk on väiksem kui k:
   - valime ühe sellise tipu ja paneme ta magasini;
   - kustutame selle tipu ja temaga seotud servad graafist.
2. Kui graaf on mittetühi (ja kõigi tippude järk on vähemalt k), siis:
   - valime tipu (kasutades mingit heuristikat) ja pillame ta mällu;
   - kustutame selle tipu ja temaga seotud servad graafist;
   - kui selle tulemusena tekkis tippe mille järk on k-st väiksem, siis lähme sammule 1;
   - vastasel korral jätkame sammuga 2.
3. Võtame üksteise järel magasinist tippe ja värvime nad vähima värviga, mis erineb juba värvitud naabertippude värvidest.

Chaitini algoritm

Näide:

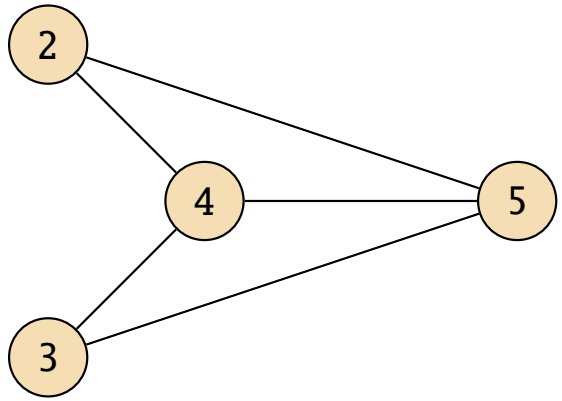
Stack



Chaitini algoritm

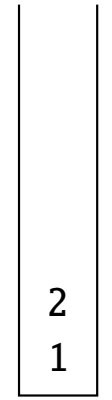
Näide:

Stack

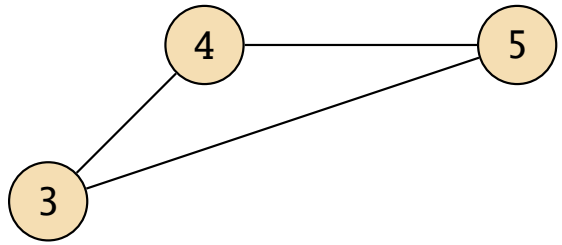


Chaitini algoritm

Näide:



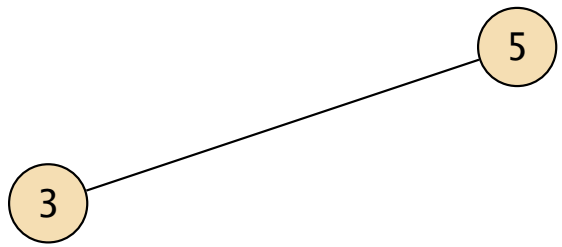
Stack



Chaitini algoritm

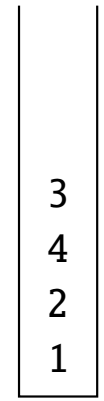
Näide:

Stack



Chaitini algoritm

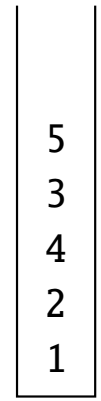
Näide:



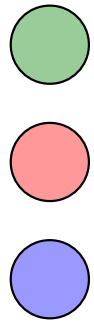
Stack

Chaitini algoritm

Näide:



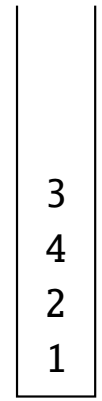
Stack



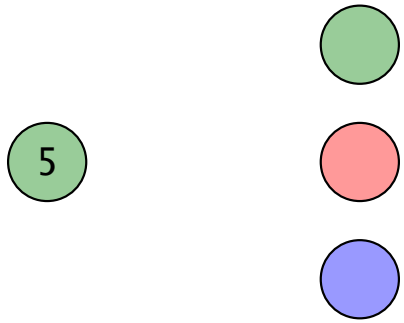
Colors

Chaitini algoritm

Näide:



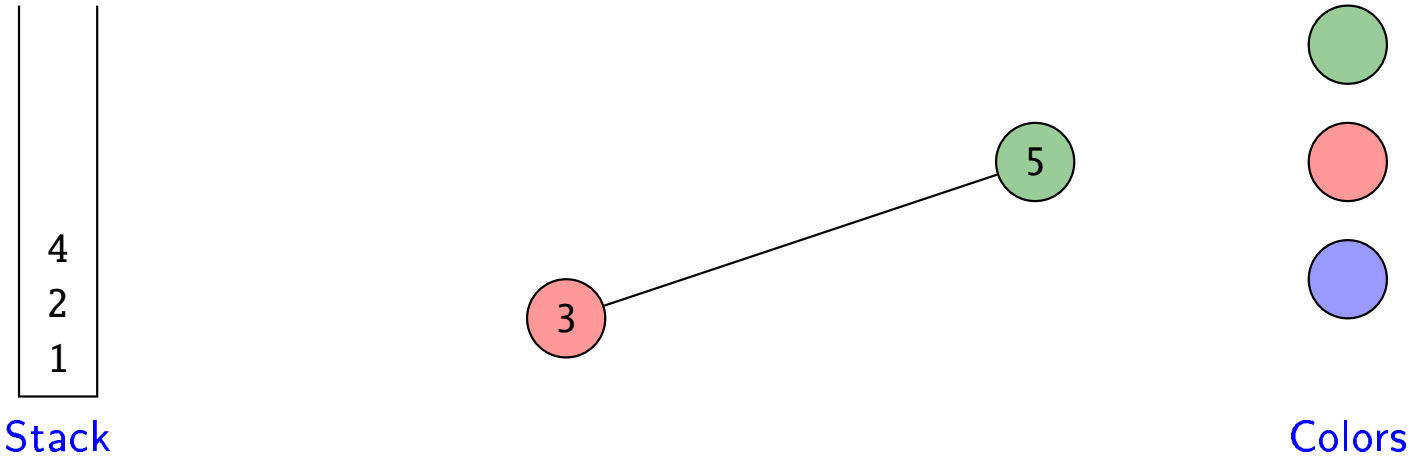
Stack



Colors

Chaitini algoritm

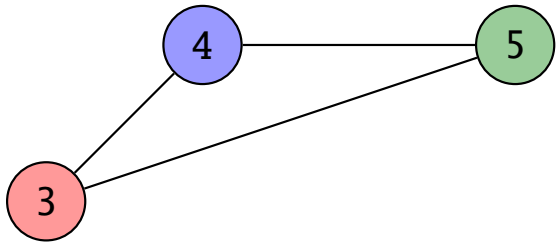
Näide:

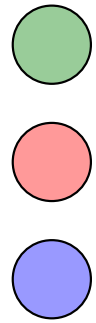


Chaitini algoritm

Näide:

Stack

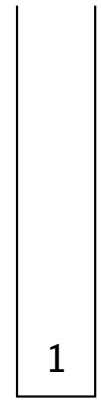




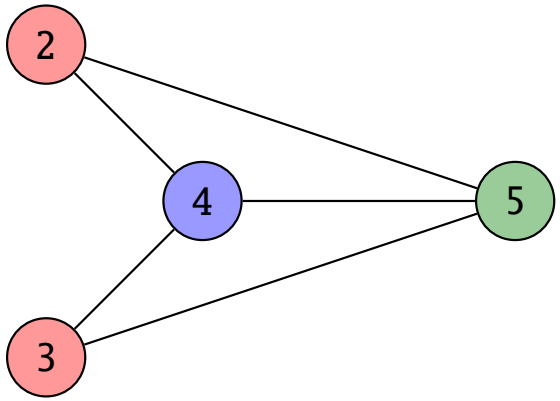
Colors

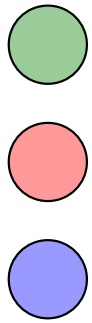
Chaitini algoritm

Näide:



Stack

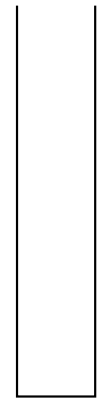




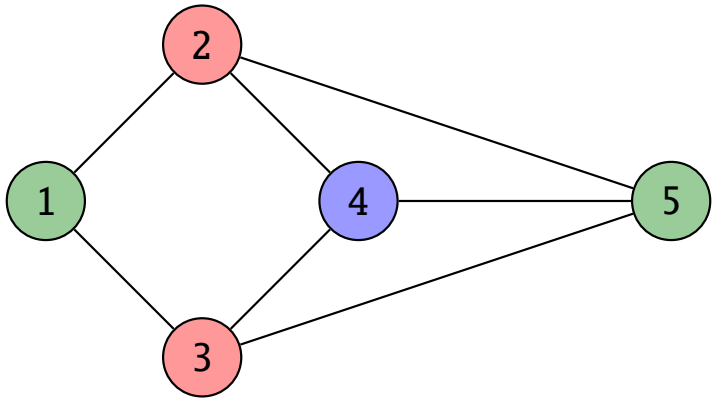
Colors

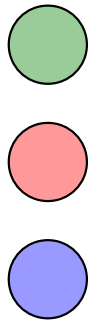
Chaitini algoritm

Näide:



Stack



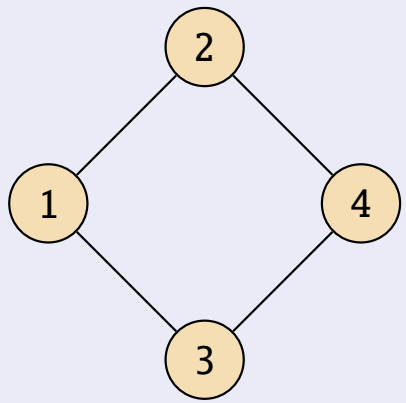


Colors

Graafi värvimine

Optimistlik värvimine (Briggs et al)

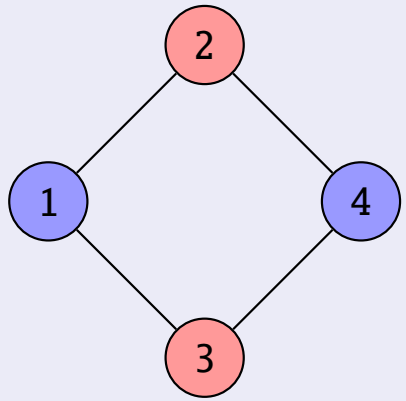
- Kui kõik tipud on järguga vähemalt k, siis mitte ei pilla tippe mällu vaid paneme mingi prioriteedi alusel magasini.
  - Magasinist tippe võttes võivad nad kõik olla ikkagi värvitavad!
- Näide alljärgnev graaf on 2-värvitav:



Graafi värvimine

Optimistlik värvimine (Briggs et al)

- Kui kõik tipud on järguga vähemalt k, siis mitte ei pilla tippe mällu vaid paneme mingi prioriteedi alusel magasini.
  - Magasinist tippe võttes võivad nad kõik olla ikkagi värvitavad!
- Näide alljärgnev graaf on 2-värvitav:



Graafi värvimine

Chaitin-Briggs'i algoritm

1. Kuni leidub tippe mille järk on väiksem kui k:
   - valime ühe sellise tipu ja paneme ta magasini;
   - kustutame selle tipu ja temaga seotud servad graafist.
2. Kui graaf on mittetühi (ja kõigi tippude järk on vähemalt k), siis:
   - valime tipu, paneme ta magasini ning kustutame ta (koos servadega) graafist;
   - kui selle tulemusena tekkis tippe mille järk on k-st väiksem, siis lähme sammule 1;
   - vastasel korral jätkame sammuga 2.
3. Võtame üksteise järel magasinist tippe ja värvime nad vähima vaba värviga.
   - Kui tippu ei õnnestu värvida, siis valime värvimata tipu, pillame ta mällu ja alustame sammult 1.

Graafi värvimine

Pillamisheuristikad

- Pillamiseks tipu valimine on efektiivsuse kohalt kriitilise tähtsusega.
- Chaitini heuristika:
  - minimiseerida pillamiskao *cost* ja tipu hetkejärgu *degree* jagatise $\frac{cost}{degree}$ väärtust;
  - so. valime pillamiseks võimalikult "odava" tipu, mis samas vähendab võimalikult paljude tippude järku.
- Alternatiivseid meetrikaid: $\frac{cost}{degree^2}$, *cost*, pillamiste arv.
- Variatsioonid:
  - interferentsipiirkonna pillamine;
  - elususpiirkondade tükeldamine;
  - rematerialiseerimine.