1 Definitionen zum Vertex Cover

1.1 Vertex Cover

Geg.: ungerichteter Graph $G = (V, E)$ mit n Knoten in der Knotenmenge $V = \{v_1, \ldots, v_n\}$ und den Kanten $E \subset V \times V$.

Ges.: Knotenmenge $C \subset V$ mit folgender Eigenschaft:

$$\forall (a, b) \in E : a \in C \vee b \in C$$

1.2 Minimum Vertex Cover

Geg.: ungerichteter Graph $G = (V, E)$ mit n Knoten.

Ges.: Vertex Cover mit minimaler Knotenzahl

$$|C| \ minimal$$

1.3 Weighted Vertex Cover

Geg.: ungerichteter Graph $G = (V, E)$ mit n Knoten. Gewicht zu jedem Knoten gegeben durch die Knotengewichtsfunktion $w : V \to \mathcal{R}^+$ bzw. $w : V \to \mathcal{N}$.

Ges.: Vertex Cover mit minimaler Knotengewichtssumme

$$\sum_{v \in C} w(v) \ minimal$$

2 Algorithmen für Weighted Vertex Cover

2.1 Randomisierter Algorithmus für Weighted Vertex Cover

2.1.1 Programm: Random Weighted Vertex Cover (RWVC)

Eingabe Graph $G = (V, E)$, Gewichtsfunktion $w : V \to \mathcal{N}$;
Ausgabe Vertex Cover U;
$U := \emptyset$
Solange $E \neq \emptyset$ **mache**
  wähle Kante $(v, t) \in E$
  wähle zufällig $x \in \{v, t\}$ mit der Wahrscheinlichkeit
    $P\{x = v\} = \frac{w(t)}{w(v)+w(t)}$
  $U := U \cup \{x\}$;
  $E := E \setminus \{e \in E : e \ ist \ inzident \ zu \ x\}$
egnaloS
Gebe zurück U

2.1.2 Approximationsfaktor von 2

Vor.: x sei Instanz von Weighted Vertex Cover
ungerichteter Graph $G = (V, E)$
Gewichtsfunktion $w : V \to \mathcal{N}$
U sei approximative Lösung, mittels Programm 2.1.1 gefunden
das Maß von U sei

$$m_{RWVC}(x) := \sum_{v \in U} w(v)$$

U^* sei die optimale Lösung, oder eine davon
das Maß von U^* sei

$$m^*(x) := \sum_{v \in U^*} w(v)$$

Beh.: $E(m_{RWVC}(x)) \leq 2m^*(x)$

Bew.: für Knoten $v \in V$ definieren wir:

$$X_v := \begin{cases} w(v) für\ v \in\ U \\ 0\ sonst \end{cases}$$

wegen $U \subset V$ gilt:

$$E(m_{RWVC}(x)) = E\left(\sum_{v\in U} X_v\right) = \sum_{v\in U} E(X_v) \le \sum_{v\in V} E(X_v)$$

wegen $U^* \cap U \subset U^*$ gilt:

$$\sum_{v\in U^*} E(X_v) = E\left(\sum_{v\in U^*} X_v\right) = E\left(\sum_{v\in U^*\cap U} w(v)\right) \le E\left(\sum_{v\in U^*} w(v)\right) = m^*(x)$$

daher genügt zu zeigen:

$$\sum_{v\in V} E(X_v) \le 2 \sum_{v\in U^*} E(X_v) \tag{1}$$

genauer: Wenn Gl. (1) erfüllt ist, dann ist auch die Beh. erfüllt. Wir werden sehen, daß Gl. (1) erfüllt ist.

$$\begin{aligned} N(v) := \ & \{t \in V : (v,t) \in E\} \cup \{t \in V : (t,v) \in E\} = \\ & \{Menge\ aller\ zu\ v\ adjazenten\ Knoten\} \end{aligned}$$

(v,t) sei erste gewählte Kante im Algorithmus. Wir definieren *wählt*:
(v,t) *wählt* im nächsten Schritt v, wenn v zufällig gewählt wurde, sonst *wählt* t.

$$X_{(v,t),\tilde{v}} := \begin{cases} w(\tilde{v})\ \ für\ \ (v,t)\ wählt\ \tilde{v} \in \{v,t\} \\ 0\ \ \ sonst \end{cases}$$

Wenn $X_{(v,t),v} = w(v)$, dann gilt: $\forall t' \in N(v) \setminus \{t\} : X_{(v,t'),v} = 0$

$$\Rightarrow X_v = \sum_{t\in N(v)} X_{(v,t),v}$$

$$\Rightarrow E(X_v) = E\left(\sum_{t\in N(v)} X_{(v,t),v}\right) = \sum_{t\in N(v)} E(X_{(v,t),v})$$

$$\begin{aligned} E\left(X_{(v,t),v}\right) \ &= \ w(v)\ P\{(v,t)\ wählt\ v\} = \\ & w(v)\ P\{(v,t)\ wurde\ gewählt\}\ \frac{w(t)}{w(v)+w(t)} = \\ & w(t)\ P\{(v,t)\ wurde\ gewählt\}\ \frac{w(v)}{w(v)+w(t)} = \\ & w(t)\ P\{(v,t)\ wählt\ t\} = \\ & E\left(X_{(v,t),t}\right) \end{aligned}$$

also

$$E\left(X_{(v,t),v}\right) = E\left(X_{(v,t),t}\right) \tag{2}$$

$$\sum_{v\in U^*} E(X_v) = \sum_{v\in U^*}\sum_{t\in N(v)} E\left(X_{(v,t),v}\right) \tag{3}$$

$$\sum_{v\notin U^*} E(X_v) = \sum_{v\notin U^*}\sum_{t\in N(v)} E\left(X_{(v,t),v}\right) \overset{(2)}{=} \sum_{v\notin U^*}\sum_{t\in N(v)} E\left(X_{(v,t),t}\right) \tag{4}$$

Hilfs-Beh.:

$$\forall v \notin U^* : \forall t \in N(v) : t \in U^*$$

Hilfs-Bew.: Ann.: $\exists v_1 \notin U^* : \exists v_2 \in N(v_2) : v_2 \notin U^*$
$\Rightarrow$ Kante (v_1, v_2) wird nicht überdeckt. Widerspruch. Hilfs-Beh. richtig.

sei $v_1 \notin U^*$ und $t_1 \in N(v_1)$, dann gilt

- nach Hilfs-Beh.: $t_1 \in U^*$
- offensichtlich: $v_1 \in N(t_1)$
- weiterhin gilt wegen ungerichteter Graph: $(v_1, t_1) = (t_1, v_1)$

d. h.: Für jeden Term $E(X_{(v,t),t}) := E(X_{(v_1,t_1),t_1})$ in Gl. (4), muß daher in Gl. (3) ein entsprechender Term $E(X_{(v,t),v}) = E(X_{(t_1,v_1),t_1})$ vorkommen.
Dies impliziert die
Hilfs-Beh.:

$$\sum_{v \notin U^*} E(X_v) \leq \sum_{v \in U^*} E(X_v)$$

Hilfs-Bew.:

$$\text{Ann.:} \sum_{v \notin U^*} E(X_v) \quad > \quad \sum_{v \in U^*} E(X_v)$$

$$\Rightarrow \sum_{v \notin U^*} \sum_{t \in N(v)} E\left(X_{(v,t),t}\right) \quad > \quad \sum_{v \in U^*} \sum_{t \in N(v)} E\left(X_{(v,t),v}\right)$$

$$\Rightarrow \exists v_1 \notin U^* : \exists t_1 \in N(v_1) : \nexists v_2 \in U^* : \exists t_2 \in N(v_2) : E\left(X_{(v_1,t_1),v_1}\right) = E\left(X_{(v_2,t_2),v_2}\right)$$

Widerspruch. Hilfs-Beh. richtig.

$$\sum_{v \in V} E(X_v) = \sum_{v \in U^*} E(X_v) + \sum_{v \notin U^*} E(X_v) \leq 2 \sum_{v \in U^*} E(X_v)$$

Somit ist Gl. (1) erfüllt und die Beh. gezeigt.

2.2 Nicht-Randomisierter Algorithmus für Weighted Vertex Cover

2.2.1 Programm: Weighted Vertex Cover (WVC)

Eingabe Graph $G = (V, E)$, Gewichtsfunktion $w : V \to \mathcal{N}$;
Ausgabe Vertex Cover C;
$C := \emptyset$
Solange $V \neq \emptyset$ **mache**
  Solange $\exists x \in V : \forall v \in V : (x, v) \notin E$ **mache**
    $V := V \setminus \{x\}$
  egnaloS
  wähle $v \in V$ mit minimalem $\frac{w(v)}{Grad(v)}$
  $C := C \cup \{v\}$
  Solange $\exists u \in N(v)$ **mache**
    $E := E \setminus \{(u, v)\}$
    $w(u) := w(u) - \frac{w(v)}{Grad(v)}$
  egnaloS
egnaloS
Gebe zurück U

2.2.2 Approximationsfaktor von 2

Vor.: x sei Instanz von Weighted Vertex Cover
ungerichteter Graph $G = (V, E)$
Gewichtsfunktion $w : V \to \mathcal{N}$
U sei approximative Lösung, mittels Programm 2.2.1 gefunden
das Maß von U sei

$$m_{WVC}(x) := \sum_{v \in U} w(v)$$

U^* sei die optimale Lösung, oder eine davon
das Maß von U^* sei

$$m^*(x) := \sum_{v \in U^*} w(v)$$

Beh.: $m_{WVC}(x) \leq 2m^*(x)$

3 Definitionen zur Satisfizierbarkeit (satisfiability)

$\tau = \{X_i : i \in \mathcal{N}\}$ ist die abzählbare Menge von Aussagevariablen.
Die Menge AL der Formeln der Aussagenlogik ist wie folgt rekursiv gegeben:

- $\tau \subset AL$
- $\psi,\ \varphi \in AL \Rightarrow (\psi \wedge \varphi),\ (\psi \vee \varphi),\ \neg\psi,\ \neg\varphi\ \in AL$

τ_ψ ist die Menge der Aussagevariablen, die in der Formel $\psi \in AL$ vorkommen.
Die *Belegung* einer Formel $\psi \in AL$ ist $f : \tau_\psi \to \{0,1\}$.
Der *Wahrheitswert* einer Formel bei der Belegung f ist $I_f : AL \to \{0,1\} = \{falsch, wahr\}$ und rekursiv gegeben durch:

- für $\psi \in \tau$ ist $I_f(\psi) = f(\psi)$ durch die Belegung definiert
- $I_f(\psi \wedge \varphi) = min\{I_f(\psi), I_f(\varphi)\}$
- $I_f(\psi \vee \varphi) = max\{I_f(\psi), I_f(\varphi)\}$
- $I_f(\neg\psi) = 1 - I_f(\psi)$

$\psi \in AL$ heißt *erfüllbar*, wenn es eine Belegung $f : \tau_\psi \to \{0,1\}$ von ψ gibt, so daß $I_f(\psi) = 1$.
$\psi \in AL$ heißt *Tautologie*, wenn für jede Belegung f von ψ gilt $I_f(\psi) = 1$.
Ein *Literal* ist eine Aussagenvariable X_i oder deren Negat $\neg X_i$.
Eine Formel $\psi \in AL$ ist in *konjunktiver Normalform (KNF)*, wenn sie die Form

$$\psi := \bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} Y_{ij}$$

hat, wobei die Y_{ij} Literale sind.
Eine Formel $\psi \in AL$ ist in *disjunktiver Normalform (DNF)*, wenn sie die Form

$$\psi := \bigvee_{i=1}^{n} \bigwedge_{j=1}^{m_i} Y_{ij}$$

hat, wobei die Y_{ij} Literale sind.
Eine Formel $\varphi \in AL$ wird *Klausel* genannt, wenn sie die Form

$$\varphi := \bigvee_{i} Y_i$$

hat, wobei die Y_i Literale sind.

3.1 Maximum Satisfiability

Geg.: $V = \{v_1, \ldots, v_n\}$ Menge von n Aussagevariablen.
$C = \{c_1, \ldots, c_t\}$ Menge von t Klauseln, wobei jede Klausel c_j aus r_j Variablen aus V gebildet wird.

Ges.: Belegung $f : V \to \{0;1\}$ der n Aussagevariablen, so daß maximale Anzahl von Klauseln erfüllt ist

$$\sum_{c_j \in C} I_f(c_j)\ maximal$$

3.2 Maximum Weighted Satisfiability

Geg.: $V = \{v_1, \ldots, v_n\}$ Menge von n Aussagevariablen.
$C = \{c_1, \ldots, c_t\}$ Menge von t Klauseln, wobei jede Klausel c_j aus r_j Variablen aus V gebildet wird.
Gewichtsfunktion $w : C \to R^+$

Ges.: Belegung $f : V \to \{0,1\}$ der n Aussagevariablen, so daß die gewichtete Summe der erfüllten Klauseln maximal ist

$$\sum_{c_j \in C} w(c_j) I_f(c_j)\ maximal$$

4 Randomisierter Algorithmus für Maximum (Weighted) Satisfiability

4.1 Programm: Random Satisfiability (RS)

Eingabe Menge C von Klauseln auf einer Menge von Variablen V
Ausgabe Belegung $f : V \to \{0,1\}$
für jedes $v \in V$:
setze unabhängig voneinander $f(v) = 1$ mit Wahrscheinlichkeit $\frac{1}{2}$
Gebe zurück f

4.2 Approximationsfaktor

Vor.: x sei Instanz von Maximum Satisfiability
$V = \{v_1, \ldots, v_n\}$ sei Menge der n Aussagevariablen
$C = \{c_1, \ldots, c_t\}$ Menge von t Klauseln aus r_j Variablen, wobei $\forall j : r_j \geq k$
$f : V \to \{0,1\}$ sei approximative Lösung, mittels Programm 4.1 gefunden
das Maß von f sei

$$m_{RS}(x) := \sum_{c_j \in C} I_f(c_j)$$

Beh.: $E(m_{RS}(x)) \geq \left(1 - 2^{-k}\right) t$

Bew.: t ist das Maß der Lösung, wenn alle t Klauseln erfüllt sind.
Die Wahrscheinlichkeit, daß eine Klausel c_j aus k Literalen durch die Belegung f nicht erfüllt ist, ist

$$P\left\{\begin{matrix} Klausel\ c_j\ aus\ k\ Literalen\ ist \\ durch\ Belegung\ f\ nicht\ erf\ddot{u}llt \end{matrix}\right\} = \left(\frac{1}{2}\right)^k$$
$$= P\left\{\begin{matrix} alle\ Literale\ in\ c_j\ werden \\ durch\ f\ auf\ 0\ gesetzt \end{matrix}\right\}.$$

$$P\left\{\begin{matrix} Klausel\ c_j\ aus\ k\ Literalen \\ ist\ durch\ Belegung\ f\ erf\ddot{u}llt \end{matrix}\right\} = 1 - 2^{-k}$$

Summierung liefert:

$$E(m_{RS}(x)) \geq \left(1 - \frac{1}{2^k}\right) t$$

Bem.: Das $\geq$ kommt aus der Abschätzung $r_j \geq k$.
Mit der optimalen Lösung $m^*(x)$, wobei $m^*(x) \leq t$, ergibt obige Aussage:

$$\frac{m^*(x)}{E(m_{RS}(x))} \leq \frac{1}{1 - \frac{1}{2^k}}$$

Ohne Einschränkung der Anzahl von Literalen pro Klausel, d. h. k=1, ergibt sich:

$$\frac{m^*(x)}{E(m_{RS}(x))} \leq 2$$

für $k \geq 2$:

$$\frac{m^*(x)}{E(m_{RS}(x))} \leq \frac{4}{3}$$

4.3 Approximationsfaktor beim erweiterten Problem

Vor.: x sei Instanz von Maximum Weighted Satisfiability
$V = \{v_1, \ldots, v_n\}$ sei Menge der n Aussagevariablen
$C = \{c_1, \ldots, c_t\}$ Menge von t Klauseln aus r_j Variablen, wobei $\forall j : r_j \geq k$
$w : C \to R^+$ sei Gewichtsfunktion
$f : V \to \{0,1\}$ sei approximative Lösung, mittels Programm 4.1 gefunden
das Maß von f sei

$$m_{RWS}(x) := \sum_{c_j \in C} w(c_j) I_f(c_j)$$

Beh.:

$$E(m_{RWS}(x)) \geq \left(1 - 2^{-k}\right) \sum_{c_j \in C} w(c_j)$$

Bew.: wie unter 4.2

Bem.: Zum Lösen von Maximum Weighted Satisfiability wurde die Gewichtung der Klauseln nicht berücksichtigt.
Für die optimale Lösung gilt

$$m^*(x) \leq \sum_{c_j \in C} w(c_j) \tag{5}$$

Somit läßt sich die gleiche Abschätzung wie unter 4.2 Bem. wiederholen:

$$E(m_{RWS}(x)) \geq \left(1 - 2^{-k}\right) \sum_{c_j \in C} w(c_j) \overset{(5)}{\geq} \left(1 - 2^{-k}\right) m^*(x)$$

Es gilt also die gleiche Abschätzung für den Approximationsfaktor.

5 Randomisierter Algorithmus für Maximum Weighted Satisfiability

x sei Instanz von Maximum Weighted Satisfiability, wie unter 3.2 definiert.

5.1 Programm: General Random Weighted Satisfiability 1. Fassung (GRWS)

Eingabe Menge C von Klauseln auf einer Menge von Variablen V, Gewichtsfunktion $w : C \to N$
Ausgabe Belegung $f : V \to \{0, 1\}$
für jedes $v_i \in V$:
  setze $f(v_i) = 1$ mit der Wahrscheinlichkeit p_i
Gebe zurück f

5.2 Erste Analyse von GRWS (Programm 5.1)

Vor.: $m_{GRWS}(x)$ sei das Maß der Lösung mittels Programm 5.1 gefunden.

$$\begin{aligned}
V_c &:= \{i \in N : v_i \in V \text{ kommt in c vor}\} = \\
&\quad \{i \in N : v_i \in \tau_c\} \\
V_c^+ &:= \{i \in V_c : v_i \text{ kommt nichtnegiert in c vor}\} = \\
&\quad \{i \in V_c : (I_f(v_i) = 1 \Rightarrow I_f(c) = 1)\} \\
V_c^- &:= \{i \in V_c : v_i \text{ kommt negiert in c vor}\} = \\
&\quad \{i \in V_c : (I_f(v_i) = 0 \Rightarrow I_f(c) = 1)\}
\end{aligned}$$

dann gilt:

$$E(m_{GRWS}(x)) = \sum_{c \in C} w(c) \overbrace{(1 - \underbrace{\prod_{i \in V_c^+} (1-p_i) \prod_{i \in V_c^-} p_i}_{\substack{Wahrscheinlichkeit: \\ c\ nicht\ erfüllt}})}^{Wahrscheinlichkeit:\ c\ erfüllt}$$

5.3 Programm: Integer Linear Program IP-SAT(x)

$$maximiere\ m_{IP-SAT}(x) := \sum_{c_j \in C} w(c_j) z_j$$

$$wobei:\ \forall c_j \in C: \sum_{i \in V_{c_j}^+} y_i + \sum_{i \in V_{c_j}^-} (1-y_i) \geq z_j$$

$$mit: \begin{array}{l} für\ 1 \leq i \leq n : y_i \in \{0;1\} \\ für\ 1 \leq j \leq t : z_j \in \{0;1\} \end{array} \tag{6}$$

Eine Lösung f von x korrespondiert direkt mit einer Lösung von IP-SAT(x):

- $y_i = 1 \Leftrightarrow$ Variable v_i ist wahr $\Leftrightarrow I_f(v_i) = 1$
- $z_j = 1 \Leftrightarrow$ Klausel c_j ist erfüllt $\Leftrightarrow I_f(c_j) = 1$

Durch Ausdehnung der Nebenbedingungen in IP-SAT(x), d. h. in Gl. (6) werden die diskreten Mengen durch Intervalle ersetzt, entsteht:

5.4 Programm: Linear Program LP-SAT(x)

$$maximiere\ m_{IP-SAT}(x) := \sum_{c_j \in C} w(c_j) z_j$$

$$wobei:\ \forall c_j \in C: \sum_{i \in V_{c_j}^+} y_i + \sum_{i \in V_{c_j}^-} (1-y_i) \geq z_j$$

$$\forall y_i : 0 \leq y_i \leq 1$$

$$\forall z_i : 0 \leq z_i \leq 1$$

Offensichtlich gilt für die optimale Lösung $m^*_{\dots}(x)$ von LP-SAT(x) und IP-SAT(x):

$$m^*_{LP-SAT} \geq m^*_{IP-SAT}(x)$$

5.5 Programm: General Random Weighted Satisfiability 2. Fassung (GRWS)

Eingabe Menge C von Klauseln auf einer Menge von Variablen V, Gewichtsfunktion $w : C \rightarrow N$
Ausgabe Belegung $f : V \rightarrow \{0, 1\}$
Finde Optimum (y^*, z^*) von LP-SAT(x)
für jedes $v_i \in V$:
  $p_i := g(y_i^*)$
  setze $f(v_i) = 1$ mit der Wahrscheinlichkeit p_i
Gebe zurück f

5.6 Zweite Analyse von GRWS (Programm 5.5)

Programm 5.5 läuft in polynomieller Zeit, wenn:

- LP-SAT(x) in polynomieller Zeit gelöst wird.
- g in polynomieller Zeit berechnet werden kann.

$$\text{Ann.: } \exists\alpha \; mit \; 0 < \alpha < 1 : \forall c_j \in C : 1 - \prod_{i \in V_{c_j}^+} (1 - p_i) \prod_{i \in V_c^-} p_i \geq \alpha z_j^*$$

Weiterhin gilt

$$m^*_{LP-SAT}(x) = \sum_{j=1}^{t} w(c_j) z_j^*$$

und

$$E(m_{GRWS}(x)) = \sum_{c \in C} w(c) \left(1 - \prod_{i \in V_c^+} (1 - p_i) \prod_{i \in V_c^-} p_i \right)$$

$\Rightarrow$ Programm 5.5 hat im Erwartungswert eine Approximationsfaktor von $\frac{1}{\alpha}$:

$$E(m_{GRWS}(x)) \geq \sum_{c \in C} w(c) \alpha z_j^* = \alpha m^*(x)$$

Die Funktion g ist noch nicht gewählt. Eine einfache Wahl ergibt sich:

$$\forall i : g(y_i) := y_i^*$$

5.7 Abschätzung

Vor.: x sei Instanz von Maximum Weighted Satisfiability
(y^*, z^*) sei optimale Lösung von LP-SAT(x)
$\alpha_k = 1 - (1 - \frac{1}{k})^k$

Beh.: $\forall c_j \in C$ mit $r_j = k$ Literalen gilt:

$$1 - \prod_{i \in V_c^+} (1 - y_i^*) \prod_{i \in V_c^-} y_i^* \geq \alpha_k z_j^*$$

Bew.: o.B.d.A.[1]: $\forall c_j : V_{c_j}^- = \emptyset$
Es bleibt daher zu zeigen:

$$1 - \prod_{i=1}^{k} (1 - y_{j_i}^*) \geq \alpha_k z_j^*$$

[1]Jedes Literal in c_j, das die Negation einer Aussagenvariable X_i ist, kann durch die Hinzunahme der Aussagenvariable $\neg X_i$ durch diese Aussagenvariable ersetzt werden. Voraussetung dafür ist, daß jedes Literal in der Klausel c_j, in dieser Klausel nicht zusätzlich negiert vorkommt. Dies macht allerdings in der Anwendung auch keinen Sinn.

Der Doppelindex j_i deutet die Umindizierung an.
Allgemein gilt:

$$\forall a_1, \ldots, a_k \in R^+ : \frac{1}{k} \sum_{j=1}^{k} a_j \geq \sqrt[k]{\prod_{j=1}^{k} a_j} \tag{7}$$

Bedingung von LP-SAT(x) überträgt sich zu:

$$\sum_{i=1}^{k} y_{j_i}^* \geq z_j^* \tag{8}$$

$$1 - \prod_{i=1}^{k} (1 - y_{j_i}^*) \overset{(7)}{\geq} 1 - \left(\frac{1}{k} \sum_{i=1}^{k} (1 - y_{j_i}^*) \right)^k =$$

$$1 - \left(\frac{k}{k} - \frac{1}{k} \sum_{i=1}^{k} y_{j_i}^* \right)^k \overset{(8)}{\geq} 1 - \left(1 - \frac{z_j^*}{k} \right)^k \overset{(9),(10)}{\geq} \alpha_k z_j^*$$

$$\left.\begin{array}{l} g(z_j^*) := 1 - \left(1 - \frac{z_j^*}{k}\right)^k \text{ ist im Intervall } [0;1] \text{ konvex von oben} \\ \text{mit Extrema am Rand } g(0) = 0 \text{ und } g(1) = 1 - \left(1 - \frac{1}{k}\right)^k = \alpha_k. \end{array}\right\} \tag{9}$$

$$\left.\begin{array}{c} \text{Eine Linearisierung durch diese beiden Punkte} \\ \text{liefert } g(z_j^*) \geq \alpha_k z_j^* \text{ im Intervall } [0;1]. \end{array}\right\} \tag{10}$$

Damit ist die Beh. gezeigt.

5.8 Approximationsfaktor

Vor.: x sei Instanz von Maximum Weighted Satisfiability, wobei $\forall j$: $r_j \leq k$
$\alpha_k := 1 - \left(1 - \frac{1}{k}\right)^k$
$f : V \to \{0;1\}$ sei approximative Lösung, mittels Programm 5.5 mit $g(y_i^*) := y_i^*$ gefunden.
das Maß von f sei

$$m_{GRWS}(x) := \sum_{c_j \in C} w(c_j) I_f(c_j)$$

$m^*(x)$ sei das Maß der optimalen Lösung

Beh.:

$$\frac{m^*(x)}{E(m_{GRWS}(x))} \leq \frac{1}{\alpha_k}$$

Bew.: Unter Beachtung, daß α_k bzgl. k eine monoton fallende Funktion ist, liefert die Aussage von 5.7 die Beh.

Bem.: für $k \leq 2$:

$$\frac{m^*(x)}{E(m_{GRWS}(x))} \leq \frac{4}{3} \approx 1,333$$

für $k \geq 3$:

$$\lim_{k\to\infty} \alpha_k = \frac{1}{e} \qquad \text{und damit} \qquad \frac{m^*(x)}{E(m_{GRWS}(x))} \leq \frac{e}{e-1} \approx 1,582$$

5.9 Vergleich Programm 4.1 und Programm 5.5 führt zu einer Verbesserung

x sein Instanz von Maximum Weighted Satisfiability.
$m_{RWS}(x)$ sei das Maß der approximativen Lösung, mittels Programm 4.1 ohne Beachtung der Gewichtung gefunden.
$m_{GRWS}(x)$ sei das Maß der approximativen Lösung, mittels Programm 5.5 mit g als Identität gefunden.
$m^*_{LP-SAT}(x)$ sei das Maß der optimalen Lösung von LP-SAT(x).
$m^*(x)$ sei das Maß der optimalen Lösung.

$$\begin{array}{rclr} \frac{m^*(x)}{E(m_{GRWS}(x))} & \leq & \frac{4}{3} & für\ r_j \leq k \leq 2 \\ \frac{4}{3} & \geq & \frac{m^*(x)}{E(m_{RWS}(x))} & für\ 2 \leq k \leq r_j \end{array}$$

Dies impliziert:

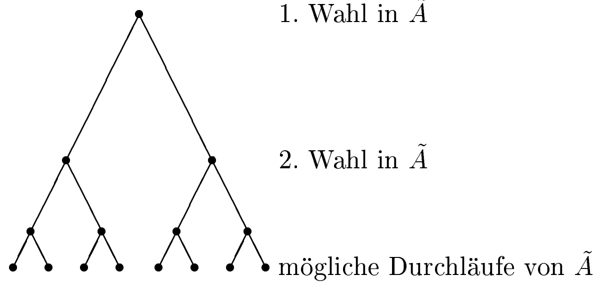
$$max\{m_{RWS}(x), m_{GRWS}(x)\} \geq \frac{m_{RWS}(x) + m_{GRWS}(x)}{2} \geq \frac{3}{4} m^*_{LP-SAT}(x) \geq \frac{3}{4} m^*(x)$$

Ein randomisierter Algorithmus der mit der gleichen Wahrscheinlichkeit $\frac{1}{2}$ zufällig Programm 4.1 oder Programm 5.5 zum Lösen von x verwendet, hat im Erwartungswert einen Approximationsfaktor von höchstens $\frac{4}{3}$.

6 Derandomisierung: Die Methode der bedingten Wahrscheinlichkeit

Ein randomisierter Algorithmus kann in manchen Fällen derandomiesiert werden, d. h. aus einem randomisierten Algorithmus kann ein derterministischer Algorithmus abgleitet werden - mit ähnlicher Laufzeit und mit einem Approximationsfaktor, der dem Erwartungswert des randomisierten Algorithmus entspricht.
Wir werden Programm 4.1 mit der Methode der bedingten Wahrscheinlichkeit derandomisieren:

$\tilde{A}$ sei randomisierter Algorithmus (hier: Programm 4.1)
x sei Instanz (hier: Maximum Weighted Satisfiability)
$r(|x|)$ seien in A unabhängige Zufallswahlen mit exakt 2 möglichen Ausgaben: 0 und 1
Dies ergibt einen Binärbaum der Höhe $r(|x|)$. Dabei bedeutet jedes linke Kind "0 wurde gewählt" und jedes rechte Kind "1 wurde gewählt":



Zu jedem Knoten u der Höhe i sei $\sigma(u)$ ein binärer String der Länge $i-1$, der die bisherige Wahl des Algorithmus $\tilde{A}$ repräsentiert. Zu jedem Blatt l sei m_l das Maß dieser Lösung des Algorithmus $\tilde{A}$. Zu jedem inneren Knoten u sei E(u) das durschnittliche Maß der Blätter, des Unterbaumes mit u als Wurzel. E(u) ist also das erwatete Maß der Lösung von $\tilde{A}$ bei Eingabe von x, wobei die ersten $|\sigma(u)|$ Wahlen genau auf $\sigma(u)$ fielen.
v, w seien 2 Kinder von u, dann gilt:

$$E(v) \geq E(U) \qquad \bigvee \qquad E(w) \geq E(u)$$

Die Derandomisierung basiert dann darauf, daß von den Knoten u zu dem Kind v oder w gegangen wird, welches das größere zu erwartende Maß besitzt.
Konkret:
x sei Instanz von Maximum Weighted Satisfiability
$v_1, \dots, v_n$ seien die boolschen Variablen in x, d. h. $v_i \in \{0; 1\}$
Programm 4.1 setzt also n-mal zufällig eine Variable v_i auf 0 oder 1.
Derandomisierung:

Im i-ten Schritt sind $\bar{v}_1, \dots, \bar{v}_{i-1}$ für $v_1, \dots, v_{i-1}$ gewählt.
$m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1}})$ sei das Zufallsmaß das Programm 4.1 findet, nachdem $\bar{v}_1, \dots, \bar{v}_{i-1}$ für $v_1, \dots, v_{i-1}$ gewählt wurde.
Nun ist v_i so zu setzen, daß $E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},v_i}))$ maximiert wird:
wenn $E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},0})) \leq E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},1}))$
  dann setze $v_i = 1$
  sonst setze $v_i = 0$

Nach n Schritten liefert dieser Algorithmus A das Maß $m_A(x)$:

$$m_A(x) = E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_n}))$$

Wie gezeigt, gilt $E(m_{RWS}(x)) \geq m^*(x)/2$, wobei $m^*(x)$ das Optimum ist; also:

$$\begin{aligned}\frac{m^*(x)}{2} &\leq E(m_{RWS}(x)) \\ &\leq E(m_{RWS}(x|_{\bar{v}_1})) \\ &\leq E(m_{RWS}(x|_{\bar{v}_1,\bar{v}_2})) \\ &\leq \dots \\ &\leq E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_n})) \;=\; m_A(x)\end{aligned}$$

Knackpunkt: Kann $E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},0}))$ bzw. $E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},1}))$ in polynomieller Zeit deterministisch berechnet werden?
x besteht aus t Klauseln $c_1, \dots, c_t$

$$E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},1})) = \sum_{j=1}^{t} w(c_j) P\{c_j \text{ ist erfüllt } |_{\bar{v}_1,\dots,\bar{v}_{i-1},1}\}$$

wobei $P\{c_j \text{ ist erfüllt } |_{\bar{v}_1,\dots,\bar{v}_{i-1},1}\}$ die Wahrscheinlichkeit ist, mit der c_j durch die Wahl von $v_{i+1}, \dots, v_n$ erfüllt wird, unter der Einschränkung, das $v_1, \dots, v_{i-1}$ mit $\bar{v}_1, \dots, \bar{v}_{i-1}$ belegt wurde.
W_i sei die Summe der Gewichte von den Klauseln, die durch die Belegung von $v_1, \dots, v_{i-1}$ mit $\bar{v}_1, \dots, \bar{v}_{i-1}$ erfüllt sind.
$C^-(i)$ sei die Menge der Klauseln, die durch diese Belegung mit $\bar{v}_1, \dots, \bar{v}_{i-1}$ nicht erfüllt sind, aber durch geeignete Wahl der $v_i, \dots, v_n$ erfüllt werden könnten.

sei $c_j \in C^-(i)$:
  wenn die Variable v_i die Klausel c_j erfüllt,
    dann $P\{c_j \text{ ist erfüllt } |_{\bar{v}_1,\dots,\bar{v}_{i-1},1}\} = 1$
    sonst (d. h. wenn die Variable v_i die Klausel c_j nicht erfüllt oder nicht vorkommt)
      sei r_j die Anzahl der Variablen die in c_j vorkommen, die von $v_1, \dots, v_i$ verschieden sind.
      Die Wahrscheinlichkeit, das eine zufällige Belegung der $v_{i+1}, \dots, v_n$ die Klausel c_j erfüllt, ist dann:

$$P\{c_j \text{ ist erfüllt } |_{\bar{v}_1,\dots,\bar{v}_{i-1},1}\} = 1 - \frac{1}{2^{r_j}}$$

Summierung über alle Klauseln ergibt:

$$E(m_{RWS}(x|_{\bar{v}_1,\dots,\bar{v}_{i-1},1})) = W_i + \sum_{\substack{c_j \in C^-(i) \\ v_i \text{ erfüllt } c_j}} 1 + \sum_{\substack{c_j \in C^-(i) \\ v_i \text{ erfüllt nicht } c_j}} \left(1 - \frac{1}{2^{r_j}}\right)$$

Dies kann in polynomieller Zeit berechnet werden.

Inhaltsverzeichnis

1 Definitionen zum Vertex Cover 1

1.1 Vertex Cover . . . 1

1.2 Minimum Vertex Cover . . . 1

1.3 Weighted Vertex Cover . . . 1

2 Algorithmen für Weighted Vertex Cover 1

2.1 Randomisierter Algorithmus für Weighted Vertex Cover . . . 1

2.1.1 Programm: Random Weighted Vertex Cover (RWVC) 1

2.1.2 Approximationsfaktor von 2 . . . 1

2.2 Nicht-Randomisierter Algorithmus für Weighted Vertex Cover . . . 3

2.2.1 Programm: Weighted Vertex Cover (WVC) . . . 3

2.2.2 Approximationsfaktor von 2 . . . 3

3 Definitionen zur Satisfizierbarkeit (satisfiability) 4

3.1 Maximum Satisfiability . . . 4

3.2 Maximum Weighted Satisfiability . . . 4

4 Randomisierter Algorithmus für Maximum (Weighted) Satisfiability 5

4.1 Programm: Random Satisfiability (RS) . . . 5

4.2 Approximationsfaktor . . . 5

4.3 Approximationsfaktor beim erweiterten Problem . . . 5

5 Randomisierter Algorithmus für Maximum Weighted Satisfiability 6

5.1 Programm: General Random Weighted Satisfiability 1. Fassung (GRWS) . . . 6

5.2 Erste Analyse von GRWS (Programm 5.1) . . . 6

5.3 Programm: Integer Linear Program IP-SAT(x) . . . 7

5.4 Programm: Linear Program LP-SAT(x) . . . 7

5.5 Programm: General Random Weighted Satisfiability 2. Fassung (GRWS) . . . 7

5.6 Zweite Analyse von GRWS (Programm 5.5) . . . 7

5.7 Abschätzung . . . 8

5.8 Approximationsfaktor . . . 8

5.9 Vergleich Programm 4.1 und Programm 5.5 führt zu einer Verbesserung . . . 9

6 Derandomisierung: Die Methode der bedingten Wahrscheinlichkeit 9