Entwurf und Analyse von Algorithmen

WS 1999/2000
Prof. Dr. K. Madlener
Übungsblatt 12

Aufgabe 12.1. In dieser Aufgabe werden wir randomisierte Approximations-Algorithmen (RAA) kennenlernen. Dabei heiße ein RAA für ein Maximierungsproblem genau dann ein ϵ-RAA, wenn für das tatsächliche Maximum m_* und die Näherung $\widehat{m} \leq m_*$ (wobei $\widehat{m}$ eine Zufallsvariable ist) gilt: $(m_* - E(\widehat{m}))/m_* \leq \epsilon$. (Dabei sei $E(\cdot)$ der Erwartungswert einer Zufallsvariablen.)

Wir betrachten eine Anwendung des Randomized Rounding (RR). Gegeben sei eine aussagenlogische Formel φ in konjunktiver Normalform. Das Problem MAX-SAT ist es nun, eine Belegung der aussagenlogischen Variablen zu finden, die möglichst viele der Klauseln von φ erfüllt. (Erinnerung: Das SAT-Problem war, zu entscheiden, ob es eine Belegung gibt, die φ erfüllt, d. h. alle Klauseln von φ erfüllt, und ggf. eine solche Belegung zu konstruieren.)

Man weiß, dass MAX-SAT NP-vollständig ist. Es gilt aber folgende Aussage: Für jede aussagenlogische Formel in konjunktiver Normalform mit m Klauseln gibt es eine Belegung der aussagenlogischen Variablen, die mindestens $m/2$ Klauseln der Formel erfüllt. Zeigen Sie dies. Zeigen Sie auch, dass diese Schranke im Allgemeinen nicht verbessert werden kann. Geben Sie dann einen (1/2)-RAA für das allgemeine MAX-SAT-Problem und einen (2^{-k})-RAA für das MAX-SAT-Problem, das auf Formeln mit mindestens k (verschiedenen) Literalen je Klausel eingeschränkt ist, an.

Damit haben wir also einen (1/4)-RAA für MAX-SAT, falls jede Klausel wenigstens zwei (verschiedene) Literale hat. Begrenzender Faktor für den Approximationsfaktor im allgemeinen Fall scheinen Klauseln mit nur einem Literal zu sein. Wir werden nun einen Algorithmus kennenlernen, der speziell im Falle, dass viele Klauseln nur ein Literal besitzen, „gut“ arbeitet.

Dazu assoziieren wir zu jeder Klausel C_j der Formel eine Indikatorvariable $z_j \in \{0, 1\}$, die anzeigt, ob die Klausel erfüllt ist oder nicht. Zu jeder aussagenlogischen Variablen x_i assoziieren wir eine Indikatorvariable $y_i \in \{0, 1\}$, die genau dann 1 ist, wenn x_i zu „wahr“ ausgewertet wird. Formulieren Sie nun MAX-SAT als Instanz des Problems der ganzzahligen linearen Programmierung (ILP).

Wir wenden nun das Prinzip des RR an, indem wir die Ganzzahligkeits-Bedingungen der obigen ILP-Instanz fallenlassen und erlauben, dass die y_i und die z_j Werte in $[0, 1]$ annehmen. Seien dann $\widehat{y_i}$ und $\widehat{z_j}$ die durch Lösung der relaxierten Probleminstanz erhaltenen

Werte für y_i bzw. z_j. Wir setzen nun jede Variable y_i mit Wahrscheinlichkeit $\widehat{y_i}$ unabhängig von den anderen Variablen $\widehat{y_{i'}}$ auf 1 (was zum Setzen von x_i auf „wahr" korrespondiert).

Zeigen Sie: Wenn C_j eine Klausel mit k (verschiedenen) Literalen ist, so ist die Wahrscheinlichkeit, dass C_j durch eine solchermaßen festgelegte Belegung der aussagenlogischen Variablen erfüllt wird, mindestens $(1-(1-1/k)^k)\widehat{z_j}$. Zeigen Sie weiter, dass das geschilderte Verfahren des RR in Verbindung mit linearer Programmierung ein $(1/e)$-RAA ist (wobei e die Basis des natürlichen Logarithmus bezeichne).

Wir betrachten nun das Zusammenspiel des von Ihnen entworfenen (1/2)-RAA und des eben skizzierten $(1/e)$-RAA (basierend auf RR und linearer Programmierung). Zeigen Sie, dass die Ausführung beider Algorithmen und die darauffolgende Auswahl des besseren Ergebnisses insgesamt einen (1/4)-RAA liefert.

Aufgabe 12.2. Ziel ist es, für das 0/1-Rucksackproblem einen heuristischen Algorithmus zu entwerfen, der die Idee des Simulated Annealing benutzt.

Eine Instanz des Problems ist definiert durch Profite $p_0, \ldots, p_{n-1}$, Gewichte $w_0, \ldots, w_{n-1}$ und eine Rucksackkapazität M (alles positive ganze Zahlen). Das Lösungsuniversum ist $\mathcal{X} = \{0,1\}^n$. Ein n-Tupel $X = (x_0, \ldots, x_{n-1})$ heiße genau dann zulässig, wenn $w(X) = \sum_{i=0}^{n-1} x_i w_i \leq M$ ist; Aufgabe ist es, $\mathsf{P}(X) = \sum_{i=0}^{n-1} x_i p_i$ zu maximieren.

Wir definieren die Nachbarschaft $\mathsf{N}(X)$ von X durch $\mathsf{N}(X) = \{Y \in \mathcal{X} : \mathsf{dist}(X,Y) = 1\}$, d. h. die Nachbarschaft von X besteht aus genau den binären n-Tupeln, bei denen genau eine Stelle gegenüber X abgeändert wurde.

Der zu entwerfende Simulated-Annealing-Algorithmus wähle die jeweils abgeänderte Lösung zufällig aus der Nachbarschaft der aktuellen Lösung aus. Geben Sie einen solchen Algorithmus an und implementieren Sie ihn in einer Programmiersprache Ihrer Wahl.

Betrachten Sie folgende Probleminstanz: Profite

$$(135, 139, 149, 150, 156, 163, 173, 184, 192, 201, 210, 214, 221, 229, 240)$$

und Gewichte

$$(70, 73, 77, 80, 82, 87, 90, 94, 98, 106, 110, 113, 115, 118, 120)$$

sowie Rucksackkapazität 750. Wählen Sie als Starttemperatur $T_0 = 1000$. Wie groß ist dann zu Beginn mindestens die Wahrscheinlichkeit, dass eine Verschlechterung der aktuellen Lösung in Kauf genommen wird? Testen Sie Ihre Implementierung mit obiger Instanz und mit jeder der drei Abkühlkonstanten 0.999, 0.9995 und 0.9999 sowie jeweils mit Iterationszahlen von 1000, 5000 und 20000. Ermitteln Sie für eine gewisse Anzahl von Läufen einer jeden Testkonstellation (z. B. zehn Läufe) den minimalen, maximalen und mittleren Näherungswert für den optimalen Profit. Zählen Sie auch, wie oft Ihre Implementierung dabei den optimalen Profit von 1458 findet.

Aufgabe 12.3. Zum Thema „evolutionäre Algorithmen" wird in der Vorlesung am Freitag eine Übungsaufgabe gestellt werden, deren Bearbeitung freigestellt ist.

Informationen zur Vorlesung:
http://www-madlener.informatik.uni-kl.de/ag-madlener/lectures/eaa/ws1999-2000/eaa.html