计算智能讲义——MaxSAT 问题

计算机科学与技术学院　M201973122　李研

1 Definition

首先给出以下符号定义：

Symbol	Definition	Description
x_i	*Variable*	变元，$x_i + \overline{x}_i = 1, i = 1,2,\ldots,n$
C_i	*Clause*	子句，$C_i = \left(\bigvee_{i\in S_i^+} x_i\right) \vee \left(\bigvee_{i\in S_i^-} \overline{x}_i\right), i = 1,2,\ldots,m$
CNF	*Formula*	合取范式，$CNF = \bigwedge_{i\in S} C_i, i = 1,2,\ldots,n$

Maximum Satisfiability：找到一组 x_i 的取值，使得满足的子句数目 $\lvert C_i \rvert$ 最大。

2 Algorithms

给出以下符号定义：

Symbol	Definition	Description
$E[Z_i]$	*Expectation*	期望，对应字句 C_i 被满足的期望
$E[Z \mid x_i]$	*Conditional Expectation*	条件期望，在 x_i 确定取值的前提下，CNF 被满足的期望
$E[Z]$	*Total Expectation*	总期望，CNF 被满足的期望

2.1 Randomized Algorithm

- 算法描述：

  将 x_i 分别以 $\frac{1}{2}$ 的概率设置为 0 或 1，则 C_i 被满足的期望为 $E[Z_i] = 1 - 2^{-\lvert C_i \rvert}$，

  CNF 被满足的期望为 $E[Z] = \sum_{i=1}^{m} E[Z_i] = \sum_{i=1}^{m} \left(1 - 2^{-\lvert C_i \rvert}\right)$。

- 近似比分析：

  设 $min\left|C_i\right| = K$ ，则有

$$m\left(1-2^{-K}\right) \le \mathrm{E}[Z] \le OPT \le m$$

- 算法分析：
  - ✧ 简单粗暴，易于理解；
  - ✧ 结果不可控，近似比只是给出理论上期望的上界，而未必每次都能得到相应质量的解。

2.2 Derandomized Algorithm

- 算法描述：

  在算法①的基础上，每个变元 x_i 都有 $\frac{1}{2}$ 的概率取 0 或 1，即有 $E[Z] = \frac{1}{2}E[Z \mid x_i = 1] + \frac{1}{2}E[Z \mid x_i = 0]$。对于每个变元 x_i ，比较 $\frac{1}{2}E[Z \mid x_i = 1]$ 和 $\frac{1}{2}E[Z \mid x_i = 0]$ 的大小，选择二者中期望值较大者对应的 x_i 取值。在此基础上，进行下一步迭代。
- 近似比分析：

  因为每一步迭代都选择了期望值较大的，所以总的条件期望 $E[Z \mid x]$ 要大于随机算法的期望值 $E[Z]$：

$$\mathrm{E}[Z \mid x] \ge \mathrm{E}[Z] \ge m\left(1-2^{-K}\right)$$

- 算法分析：
  - ✧ 结果可控，在变元顺序确定的情况下能保证结果一致性；
  - ✧ 结果与变元顺序有关，没有回溯，变元的值一旦确定便不能再更改；
  - ✧ 算法复杂度较高，条件期望的计算比较耗时；
  - ✧ 可能的优化方向：使用 $branch-and-bound$ 的树搜索框架，通过维护一个全局的条件期望实现剪枝和加速搜索。

2.3 LP Rounding Algorithm

给出以下两个决策变量的定义：

Symbol	Definition	Description
y_i	$Decision\ Variables$	决策变量，变元 x_i 的 0/1 决策变量
q_i	$Decision\ Variables$	决策变量，子句 C_i 的 0/1 决策变量

- 算法描述：

  建立 MaxSAT 混合整数规划模型如下：

$$
\begin{aligned}
&\text{maximize:} && \sum_{i=1}^{m} q_i \\
&\text{s.t.} && q_i \le \sum_{j\in S_i^+} y_j + \sum_{j\in S_i^-}\left(1-y_j\right) \quad \forall i \\
& && q_i, y_j \in \{0,1\}
\end{aligned}
$$

  ✧ Relaxing：

  松弛最后一条布尔约束为$0 \le q_i, y_j \le 1$，即得到线性规划模型。

  ✧ Rounding：

  求解模型，并按照下述策略设置变元取值：

$$
\begin{aligned}
&\text{for } j=1 \text{ to } n \text{ do} \\
&\text{Independently set } x_j = \begin{cases} 1: \text{ with probability } y_j^* \\ 0: \text{ with probability } 1-y_j^* \end{cases} \\
&\text{end for}
\end{aligned}
$$

- 近似比分析：

  以$C_1 = x_1 \vee \ldots \vee x_k$为例，利用算数-几何平均不等式，得到$C_1$的部分期望如下：

$$
\begin{aligned}
\Pr[C_1] &= 1 - \prod_{j=1}^{k}(1-y_j^*) \\
&\ge 1 - \left(\frac{1}{k}\sum_{j=1}^{k}(1-y_j^*)\right)^k \\
&\ge 1 - \left(1 - \frac{q_1^*}{k}\right)^k \\
&\ge 1 - \left(1 - \frac{1}{k}\right)^k \\
&\ge q_1\left(1 - \left(1 - \frac{1}{k}\right)^k\right) \\
&\ge q_1(1 - 1/e)
\end{aligned}
$$

  则$\mathrm{E}[Z] = \sum_{i=1}^{m} \mathrm{E}\left[Z_i\right] \ge \sum_{i=1}^{m}\left(1-\left(1-\frac{1}{\left|C_i\right|}\right)^{\left|C_i\right|}\right) \ge m\left(1-\left(1-\frac{1}{K}\right)^{K}\right).$

- 算法分析：

  ✧ 结果相对可控，较算法①求解质量有较大提升；

  ✧ 求解效率与线性规划求解器的性能有较大关系，随着变元数目增多，求解时间也逐渐延长。

2.4 LP Derandomized Algorithm

- 算法描述：

  在算法③的基础上去随机化，每个变元 x_i 都有 y_i^* 的概率取 1，$1-y_i^*$ 的概率取 0，即 $E[Z]=y_i^*\cdot E[Z\mid x_i=1]+(1-y_i^*)\cdot E[Z\mid x_i=0]$。对于每个变元 x_i，比较 $y_i^*\cdot E[Z\mid x_i=1]$ 和 $(1-y_i^*)\cdot E[Z\mid x_i=0]$ 的大小，选择二者中较大的期望值，取其对应的 x_i 取值。在此基础上，进行下一步迭代。

- 近似比分析：

  同理，该算法的条件期望 $E[Z\mid x]$ 要大于算法③的期望值 $E[Z]$：

$$\mathrm{E}[Z\mid x]\geq \mathrm{E}[Z]\geq m\left(1-\left(1-\frac{1}{K}\right)^{K}\right)$$

- 算法分析：
  - ✧ 结果可控，且与变元顺序无关；
  - ✧ 继承了算法②③的优缺点，既是四类算法中求解质量最好的，也是耗时最多的；
  - ✧ 可能的优化方向：将算法②和算法④结合可以给出一个 $\frac{3}{4}$ 近似比的算法（每次从两个算法中随机挑选一个执行）：

$$\mathrm{E}[Z]=\sum_{i=1}^{m}\mathrm{E}[Z_i]\geq \sum_{i=1}^{m}\frac{1}{2}\left(\left(1-2^{-|C_i|}\right)+\left(1-\left(1-\frac{1}{|C_i|}\right)^{|C_i|}\right)\right)\geq (3/4)m$$

3 Implementation

Github: https://github.com/lyandut/MyMaXSat