Branch-and-Bound

Endre Boros

26:711:653: Discrete Optimization

Spring 2019

Branch-and-Bound Algorithm

$$\begin{array}{rcl} \max\ c^T x & = & Z(IP) \\ s.t. & & \\ Ax & \leq & b \\ x & \in & \{0,1\}^n \end{array} \qquad \leq \qquad \begin{array}{rcl} \max\ c^T x & = & Z(LP) \\ s.t. & & \\ Ax & \leq & b \\ 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S,y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S,y) \in \mathcal{P}$ such that the upper **bound** $UB(S,y) = Z(LP(S,y))$ is **as large as possible**, and delete $IP(S,y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y,0)), IP(S \cup \{j\}, (y,1))\}$.
- If a better new feasible solution is found then
  - update $Zbest$, and
  - prune (delete) all subproblems $IP(S',y') \in \mathcal{P}$ for which $UB(S',y') \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step **Main**.

Branch-and-Bound Algorithm

$$\begin{array}{rrcl} \max & c^T x & = & Z(IP) \\ s.t. & & & \\ & Ax & \leq & b \\ & x & \in & \{0,1\}^n \end{array} \qquad \leq \qquad \begin{array}{rrcl} \max & c^T x & = & Z(LP) \\ s.t. & & & \\ & Ax & \leq & b \\ & 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S,y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S,y) \in \mathcal{P}$ such that the upper **bound** $UB(S,y) = Z(LP(S,y))$ is **as large as possible**, and delete $IP(S,y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y,0)), IP(S \cup \{j\}, (y,1))\}$.
- If a better new feasible solution is found then
  - update $Zbest$, and
  - prune (delete) all subproblems $IP(S', y') \in \mathcal{P}$ for which $UB(S', y') \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step **Main**.

Branch-and-Bound Algorithm

$$\begin{array}{rcl} \max\ c^T x & = & Z(IP) \\ s.t. & & \\ Ax & \leq & b \\ x & \in & \{0,1\}^n \end{array} \qquad \leq \qquad \begin{array}{rcl} \max\ c^T x & = & Z(LP) \\ s.t. & & \\ Ax & \leq & b \\ 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S,y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S,y) \in \mathcal{P}$ such that the upper **bound** $UB(S,y) = Z(LP(S,y))$ is **as large as possible**, and delete $IP(S,y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y,0)), IP(S \cup \{j\}, (y,1))\}$.
- If a better new feasible solution is found then
  - update $Zbest$, and
  - prune (delete) all subproblems $IP(S', y') \in \mathcal{P}$ for which $UB(S', y') \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step **Main**.

Branch-and-Bound Algorithm

$$\begin{array}{rcl} \max\ c^T x & = & Z(IP) \\ s.t. & & \\ Ax & \leq & b \\ x & \in & \{0,1\}^n \end{array} \quad \leq \quad \begin{array}{rcl} \max\ c^T x & = & Z(LP) \\ s.t. & & \\ Ax & \leq & b \\ 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S, y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S, y) \in \mathcal{P}$ such that the upper **bound** $UB(S, y) = Z(LP(S, y))$ is **as large as possible**, and delete $IP(S, y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y, 0)), IP(S \cup \{j\}, (y, 1))\}$.
- If a better new feasible solution is found then
  - **update** $Zbest$, and
  - prune (delete) all subproblems $IP(S, y) \in \mathcal{P}$ for which $UB(S, y) \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step Main.

Branch-and-Bound Algorithm

$$\begin{array}{rcl} \max\ c^T x & = & Z(IP) \\ s.t. & & \\ Ax & \leq & b \\ x & \in & \{0,1\}^n \end{array} \quad \leq \quad \begin{array}{rcl} \max\ c^T x & = & Z(LP) \\ s.t. & & \\ Ax & \leq & b \\ 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S,y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S,y) \in \mathcal{P}$ such that the upper **bound** $UB(S,y) = Z(LP(S,y))$ is **as large as possible**, and delete $IP(S,y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y,0)), IP(S \cup \{j\}, (y,1))\}$.
- If a better new feasible solution is found then
  - **update** $Zbest$, and
  - **prune** (delete) all subproblems $IP(S', y') \in \mathcal{P}$ for which $UB(S', y') \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step **Main**.

Branch-and-Bound Algorithm

$$\begin{array}{rrcl} \max & c^T x & = & Z(IP) \\ s.t. & & & \\ & Ax & \leq & b \\ & x & \in & \{0,1\}^n \end{array} \quad \leq \quad \begin{array}{rrcl} \max & c^T x & = & Z(LP) \\ s.t. & & & \\ & Ax & \leq & b \\ & 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S, y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S, y) \in \mathcal{P}$ such that the upper **bound** $UB(S, y) = Z(LP(S, y))$ is **as large as possible**, and delete $IP(S, y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y, 0)), IP(S \cup \{j\}, (y, 1))\}$.
- If a better new feasible solution is found then
  - **update** $Zbest$, and
  - **prune** (delete) all subproblems $IP(S', y') \in \mathcal{P}$ for which $UB(S', y') \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step Main.

Branch-and-Bound Algorithm

$$\begin{array}{rcl} \max\ c^T x & = & Z(IP) \\ s.t. & & \\ Ax & \leq & b \\ x & \in & \{0,1\}^n \end{array} \quad \leq \quad \begin{array}{rcl} \max\ c^T x & = & Z(LP) \\ s.t. & & \\ Ax & \leq & b \\ 1 \geq x & \geq & 0 \end{array}$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S, y)$ is defined by adding to IP the constraints $x_j = y_j$, $j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S, y) \in \mathcal{P}$ such that the upper **bound** $UB(S, y) = Z(LP(S, y))$ is **as large as possible**, and delete $IP(S, y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y, 0)), IP(S \cup \{j\}, (y, 1))\}$.
- If a better new feasible solution is found then
  - **update** $Zbest$, and
  - **prune** (delete) all subproblems $IP(S', y') \in \mathcal{P}$ for which $UB(S', y') \leq Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step Main.

Branch-and-Bound Algorithm

$$
\begin{array}{rcl}
\max\ c^T x & = & Z(IP) \\
s.t. & & \\
Ax & \le & b \\
x & \in & \{0,1\}^n
\end{array}
\quad \le \quad
\begin{array}{rcl}
\max\ c^T x & = & Z(LP) \\
s.t. & & \\
Ax & \le & b \\
1 \ge x & \ge & 0
\end{array}
$$

- For indices $S \subseteq [n]$ and binary assignments $y \in \{0,1\}^S$ the **subproblem** $IP(S,y)$ is defined by adding to IP the constraints $x_j = y_j,\ j \in S$.
- Initialize a list of subproblems by $\mathcal{P} = \{IP(\emptyset, ())\}$, and $Zbest = -\infty$.

Main Choose a subproblem $IP(S,y) \in \mathcal{P}$ such that the upper **bound** $UB(S,y) = Z(LP(S,y))$ is **as large as possible**, and delete $IP(S,y)$ from $\mathcal{P}$.

- **Choose a branching variable** $j \in [n] \setminus S$ and compute/update new subproblems $\mathcal{P} = \mathcal{P} \cup \{IP(S \cup \{j\}, (y,0)), IP(S \cup \{j\}, (y,1))\}$.
- If a better new feasible solution is found then
  - **update** $Zbest$, and
  - **prune** (delete) all subproblems $IP(S', y') \in \mathcal{P}$ for which $UB(S', y') \le Zbest$.
- Stop if $\mathcal{P} = \emptyset$, otherwise return to step **Main**.

Knapsack Example

$$\begin{array}{rcrcrcrcrcl} 20x_1 & + & 14x_2 & + & 25x_3 & + & 17x_4 & + & 9x_5 & \to & \max \\ 3x_1 & + & 2x_2 & + & 5x_3 & + & 4x_4 & + & 2x_5 & \leq & 13 \end{array}$$

- Sort items by nonincreasing unit values.

Knapsack Example

$$\begin{array}{rcrcrcrcrcl} 20x_1 & + & 14x_2 & + & 25x_3 & + & 17x_4 & + & 9x_5 & \rightarrow & \max \\ 3x_1 & + & 2x_2 & + & 5x_3 & + & 4x_4 & + & 2x_5 & \leq & 13 \end{array}$$

- Sort items by nonincreasing unit values.

Knapsack Example

$$\begin{array}{rcrcrcrcrcl}
14x_2 & + & 20x_1 & + & 25x_3 & + & 9x_5 & + & 17x_4 & \to & \max \\
2x_2 & + & 3x_1 & + & 5x_3 & + & 2x_5 & + & 4x_4 & \leq & 13
\end{array}$$

Knapsack Example

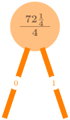
$$
\begin{array}{ccccccccccc}
14x_2 & + & 20x_1 & + & 25x_3 & + & 9x_5 & + & 17x_4 & \to & \max \\
2x_2 & + & 3x_1 & + & 5x_3 & + & 2x_5 & + & 4x_4 & \leq & 13 \\
 & & & & & & & & & & \\
1 & & 1 & & 1 & & 1 & & \mathbf{\frac{1}{4}} & & \mathbf{72\frac{1}{4}}
\end{array}
$$

Knapsack Example

$$\frac{\mathbf{72\frac{1}{4}}}{\mathbf{4}}$$

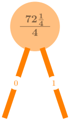
$$\begin{array}{rcrcrcrcrcl}
14x_2 & + & 20x_1 & + & 25x_3 & + & 9x_5 & + & 17x_4 & \to & \max \\
2x_2 & + & 3x_1 & + & 5x_3 & + & 2x_5 & + & 4x_4 & \leq & 13 \\
 & & & & & & & & & & \\
1 & & 1 & & 1 & & 1 & & \frac{1}{4} & & \mathbf{72\frac{1}{4}}
\end{array}$$

Knapsack Example



$$
\begin{array}{rcrcrcrcrcl}
14x_2 & + & 20x_1 & + & 25x_3 & + & 9x_5 & + & 17x_4 & \rightarrow & \max \\
2x_2 & + & 3x_1 & + & 5x_3 & + & 2x_5 & + & 4x_4 & \leq & 13 \\
 & & & & & & & & & & \\
1 & & 1 & & 1 & & 1 & & \frac{1}{4} & & \mathbf{72\,\frac{1}{4}}
\end{array}
$$

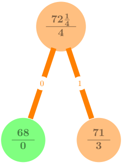
Knapsack Example



$$
\begin{array}{rcrcrcrcrcl}
14x_2 & + & 20x_1 & + & 25x_3 & + & 9x_5 & + & 17x_4 & \rightarrow & \max \\
2x_2 & + & 3x_1 & + & 5x_3 & + & 2x_5 & + & 4x_4 & \leq & 13
\end{array}
$$

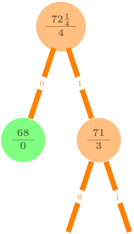
1	1	1	1	$\frac{1}{4}$	**$72\frac{1}{4}$**
1	1	1	1	**0**	**68**
1	1	$\frac{4}{5}$	0	**1**	**71**

Knapsack Example



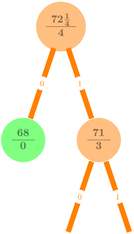
$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		$\mathbf{72\frac{1}{4}}$
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**

Knapsack Example



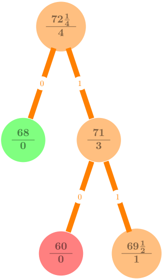
$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\to$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		**$72\frac{1}{4}$**
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**

Knapsack Example



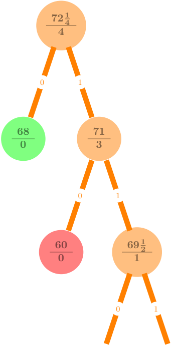
$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		**$72\frac{1}{4}$**
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		**$69\frac{1}{2}$**

Knapsack Example



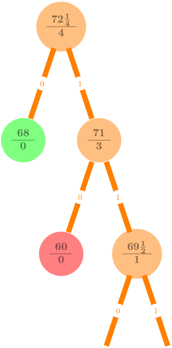
$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		**$72\frac{1}{4}$**
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		**$69\frac{1}{2}$**

Knapsack Example



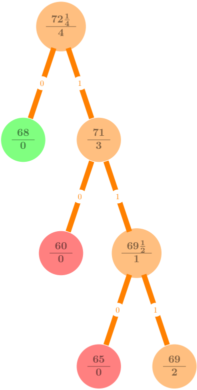
$14x_2$	+	$20x_1$	+	$25x_3$	+	$9x_5$	+	$17x_4$	$\to$	max
$2x_2$	+	$3x_1$	+	$5x_3$	+	$2x_5$	+	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		$\mathbf{72\frac{1}{4}}$
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		$\mathbf{69\frac{1}{2}}$

Knapsack Example



$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		**$72\frac{1}{4}$**
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		**$69\frac{1}{2}$**
1		**0**		**1**		1		**1**		**65**
$\frac{1}{2}$		**1**		**1**		0		**1**		**69**

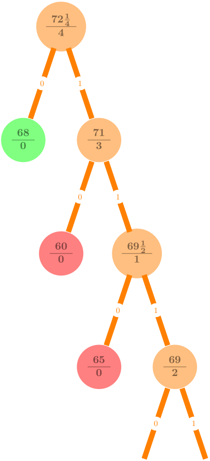
Knapsack Example



$$\begin{array}{ccccccccccc}
14x_2 & + & 20x_1 & + & 25x_3 & + & 9x_5 & + & 17x_4 & \to & \max \\
2x_2 & + & 3x_1 & + & 5x_3 & + & 2x_5 & + & 4x_4 & \leq & 13
\end{array}$$

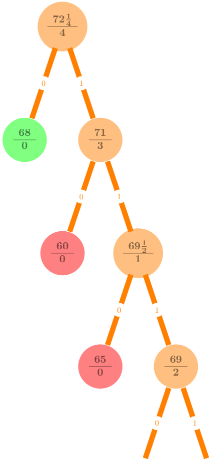
1	1	1	1	$\frac{1}{4}$	**$72\frac{1}{4}$**
1	1	1	1	**0**	**68**
1	1	$\frac{4}{5}$	0	**1**	**71**
1	1	**0**	1	**1**	**60**
1	$\frac{2}{3}$	**1**	0	**1**	**$69\frac{1}{2}$**
1	**0**	**1**	1	**1**	**65**
$\frac{1}{2}$	**1**	**1**	0	**1**	**69**

Knapsack Example



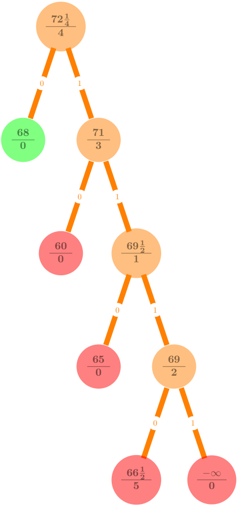
$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		**72 $\frac{1}{4}$**
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		**69 $\frac{1}{2}$**
1		**0**		**1**		1		**1**		**65**
$\frac{1}{2}$		**1**		**1**		0		**1**		**69**

Knapsack Example



$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		**$72\frac{1}{4}$**
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		**$69\frac{1}{2}$**
1		**0**		**1**		1		**1**		**65**
$\frac{1}{2}$		**1**		**1**		0		**1**		**69**
0		**1**		**1**		$\frac{1}{2}$		**1**		**$66\frac{1}{2}$**
1		**1**		**1**		–		**1**		$-\infty$

Knapsack Example



$14x_2$	$+$	$20x_1$	$+$	$25x_3$	$+$	$9x_5$	$+$	$17x_4$	$\rightarrow$	max
$2x_2$	$+$	$3x_1$	$+$	$5x_3$	$+$	$2x_5$	$+$	$4x_4$	$\leq$	13
1		1		1		1		$\frac{1}{4}$		$\mathbf{72\frac{1}{4}}$
1		1		1		1		**0**		**68**
1		1		$\frac{4}{5}$		0		**1**		**71**
1		1		**0**		1		**1**		**60**
1		$\frac{2}{3}$		**1**		0		**1**		$\mathbf{69\frac{1}{2}}$
1		**0**		**1**		1		**1**		**65**
$\frac{1}{2}$		**1**		**1**		0		**1**		**69**
0		**1**		**1**		$\frac{1}{2}$		**1**		$\mathbf{66\frac{1}{2}}$
1		**1**		**1**		$-$		**1**		$-\infty$