Sipser: Chapter 7.1 – 7.3
This game sucks and I can prove it.
Sudoku
$L_D = \{ \langle s \rangle \mid s \text{ is an instance of Sudoku } \}$
and $s$ has a solution
\[ L_D = \{ \langle s \rangle \mid s \text{ is an instance of Sudoku } \} \]
and \( s \) has a solution

\[ L_V = \{ \langle s, c \rangle \mid s \text{ is an instance of Sudoku } \} \]
and \( c \) is a solution to \( s \)
\[
L_D = \{ \langle s \rangle \mid s \text{ is an instance of Sudoku } \} \\
\times 100 \quad \text{and } s \text{ has a solution}
\]

\[
L_V = \{ \langle s, c \rangle \mid s \text{ is an instance of Sudoku } \} \\
\times 100 \quad \text{and } c \text{ is a solution to } s
\]
$L_D = 2^n$

$L_V = n^2$
$L_D = k^n$

$L_V = n^k$
\[ L_D = \{ \langle s \rangle \mid s \text{ is an instance of Sudoku} \} \]
and \( s \) has a solution

\[ L_V = \{ \langle s, c \rangle \mid s \text{ is an instance of Sudoku} \} \]
and \( c \) is a solution to \( s \)
\[ L_D = \begin{cases} 
\text{Generate the next candidate solution, } c, \text{ for } s \\
\text{If there are no more candidates, REJECT} \\
\text{Run } L_V(\langle s, c \rangle) \\
\text{If } L_V \text{ accepts, ACCEPT} 
\end{cases} \]
\[ S = \begin{cases} 
\text{Generate the next candidate solution, } c, \text{ for } s \\
\quad \text{If there are no more candidates, REJECT} \\
\text{Run } L_V(\langle s, c \rangle) \\
\quad \text{If } L_V \text{ accepts, return } c 
\end{cases} \]
\[
\begin{pmatrix}
A & B \\
C & D
\end{pmatrix}
\times
\begin{pmatrix}
E & F \\
G & H
\end{pmatrix}
=
\begin{pmatrix}
AE+BG & AF+BH \\
CE+DG & CF+DH
\end{pmatrix}
\]
$L_D = \begin{cases} 
\text{Generate the next candidate solution, } c, \text{ for } s \\
\quad \text{If there are no more candidates, REJECT} \\
\text{Run } L_V(\langle s, c \rangle) \\
\quad \text{If } L_V \text{ accepts, ACCEPT} 
\end{cases}$
Finite State Transducer

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

1 1 1
$L_D = \text{Generate candidate solutions}$

- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$
- Run $L_V(\langle s, c \rangle)$

If accept, ACCEPT

Run Time?
The class NP:

Verifiable in polynomial time

* with a deterministic TM
The class NP:

Verifiable in polynomial time

Decidable in non-deterministic polynomial time
The class \textbf{NP}:

Verifiable in polynomial time

\downarrow

Decidable in \textbf{non-deterministic} polynomial time
The class NP:

Verifiable in polynomial time

\[\downarrow\]

Decidable in non-deterministic polynomial time
The Marriage Problem

Sudoku
$$L_D = \{ \langle m \rangle \mid m \text{ is an instance of The Marriage Problem and } m \text{ has a solution} \}$$
\[ L_D = \{ \langle m \rangle \mid m \text{ is an instance of The Marriage Problem and } m \text{ has a solution} \} \]

\[ L_V = \{ \langle m, c \rangle \mid m \text{ is an instance of The Marriage Problem and } c \text{ is a solution to } m \} \]
\[
L_D = \{ \langle m \rangle \mid m \text{ is an instance of The Marriage Problem and } m \text{ has a solution} \}
\]

\[
L_V = \{ \langle m, c \rangle \mid m \text{ is an instance of The Marriage Problem and } c \text{ is a solution to } m \}
\]
\[ L_D = \begin{cases} 
\text{Generate the next candidate solution, } c, \text{ for } m \\
\text{If there are no more candidates, REJECT} \\
\text{Run } L_V(\langle m, c \rangle) \\
\text{If } L_V \text{ accepts, ACCEPT} 
\end{cases} \]
\[ S = \begin{cases} 
\text{Generate the next candidate solution, } c, \text{ for } m \\
\text{If there are no more candidates, REJECT} \\
\text{Run } L_V(\langle m, c \rangle) \\
\text{If } L_V \text{ accepts, return } c 
\end{cases} \]
\[ L_D = \text{Generate candidate solutions} \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{Run } L_V(\langle m, c \rangle) \rightarrow \text{If accept, ACCEPT} \]
\[ L_D = \{ \langle m \rangle \mid m \text{ is an instance of The Marriage Problem and } m \text{ has a solution} \} \]

Exponential Time

NP Time

\[ L_V = \{ \langle m, c \rangle \mid m \text{ is an instance of The Marriage Problem and } c \text{ is a solution to } m \} \]

Polynomial Time
Hall’s Marriage Theorem:

Suppose $G$ is a bipartite graph with partition $(A, B)$. There is a matching that covers $A$ if and only if, for every subset $X \subseteq A$, $N(X) \geq |X|$ where $N(x)$ is the number of neighbors of $X$. 
A

B

{5 4}
{6 4 2}
{5 2 1}
{4 3}
{5}
{5 1}

{1 2}
{2 3}
{4 5}
{5 6}

{1}
{2 3}
{4 5}
{2 5 6}

{3 6}
{6}
{1 4 5 6}
{1 3}
{4 5}
{2 5 6}
Hall’s Marriage Theorem:

Suppose $G$ is a bipartite graph with partition $(A, B)$. There is a matching that covers $A$ if and only if, for every subset $X \subseteq A$, $N(X) \geq |X|$ where $N(x)$ is the number of neighbors of $X$. 
Hall’s Marriage Theorem:

Suppose $G$ is a bipartite graph with partition $(A, B)$. There is a matching that covers $A$ if and only if, for every subset $X \subseteq A$, $N(X) \geq |X|$ where $N(x)$ is the number of neighbors of $X$. 
An “M-augmenting” path
An “M-augmenting” path
An “M-augmenting” path

{5, 4}
{6, 4, 2}
{5, 2, 1}
{4, 3}
{5}
{5, 2, 1}

{3, 6}
{6}
{1, 4, 5, 6}
{1, 3}
{4, 5}
{2, 5, 6}
An “M-augmenting” path
Hall’s Marriage Theorem:

**Lemma:** If $M$ is a maximum matching in graph $G$, there can be no $M$-augmenting path.

**Algo:** $M$-augmenting paths can be found in polynomial time via breadth-first search.
\[ \mathcal{L}_D = \{ \langle m \rangle \mid m \text{ is an instance of The Marriage Problem and } m \text{ has a solution in Polynomial Time} \} \]

\[ \mathcal{L}_V = \{ \langle m, c \rangle \mid m \text{ is an instance of The Marriage Problem and } c \text{ is a solution to } m \text{ in Polynomial Time} \} \]
The Marriage Problem:

1.) Verifiable in polynomial time
2.) Decidable in polynomial time

* with a deterministic TM
The class P:

1.) Verifiable in polynomial time
2.) Decidable in polynomial time

* with a deterministic TM
The class P:

1.) Verifiable in polynomial time
2.) Decidable in polynomial time
The class P:

1.) Verifiable in polynomial time
2.) Decidable in polynomial time

In NP?
The class P:

1.) Verifiable in polynomial time
2.) Decidable in polynomial time

In NP?
The Marriage Problem: NP P

Sudoku: NP
P = NP ?
The Marriage Problem

Sudoku

The Travelling Salesman
$L_D = \{ \langle t \rangle \mid t \text{ is an instance of The Travelling Salesman and } t \text{ has a solution} \}$

Yes!

$L_V = \{ \langle t, c \rangle \mid t \text{ is an instance of The Travelling Salesman and } c \text{ is a solution to } t \}$
\[ L_D = \{ \langle t \rangle \mid t \text{ is an instance of The Travelling Salesman and } t \text{ has a solution } < x \} \]

\[ L_V = \{ \langle t, c \rangle \mid t \text{ is an instance of The Travelling Salesman and } c \text{ is a solution to } t \} \]
\[ S = \begin{cases} 
    \text{Generate the next candidate solution, } c, \text{ for } t \\
    \text{If there are no more candidates, REJECT} \\
    \text{Run } L_V(\langle t, c \rangle) \\
    \text{If } L_V \text{ accepts, return } c 
\end{cases} \]
$$L_V = \begin{align*}
&\begin{cases}
&\text{Generate the next candidate solution, } c', \text{ for } t \\
&\quad \text{If there are no more candidates, ACCEPT} \\
&\quad \text{Calculate } \langle t, c' \rangle \\
&\quad \text{If } \langle t, c' \rangle \text{ is lower than } \langle t, c \rangle, \text{ REJECT}
\end{cases}
\end{align*}$$
\[ L_V = \{ \text{Generate candidate solutions} \} \]

If lower, REJECT
\[ L_D = \{ \langle t \rangle \mid t \text{ is an instance of The Travelling Salesman and } t \text{ has a solution} \} \]

\[ L_V = \{ \langle t, c \rangle \mid t \text{ is an instance of The Travelling Salesman and } c \text{ is a solution to } t \} \]
\[
S = \{ \langle t \rangle \mid \text{t is an instance of The Travelling Salesman and t has a solution} \}
\]
\[
L_V = \{ \langle t, c \rangle \mid \text{t is an instance of The Travelling Salesman and c is a solution to t} \}
\]
The class EXPTIME:

Decision problems that can be solved in exponential time.

TS: No known polynomial time verification procedure
The Marriage $\leq$ Sudoku $\leq$ The Travelling Salesman
Find the prime factors of 39203

The Marriage ≤ Sudoku ≤ The Travelling Salesman Problem
Find the prime factors of 39203