Theorem 1. *The following two statements are equivalent:*

1. $\mathcal{P} = \mathcal{NP}$.
2. *Every polytime verifiable search problem is solvable in polytime.*

Proof. The easy direction is $2 \Rightarrow 1$. Assume (2) is true, and let $L \in \mathcal{NP}$. We prove that $L \in \mathcal{P}$. Let R be the witness-relation for L, which is polytime verifiable.

By assumption, there is a polytime machine M that solves R. To decide L in polynomial time, we get x, run $M(x)$, and accept iff $M(x) \neq \perp$. This is a polytime algorithm because M is polytime, and it decides L correctly because M satisfies the following: $x \in L \Rightarrow (x, M(x)) \in R$ (in particular, $M(x) \neq \perp$) and $x \notin L \Rightarrow M(x) = \perp$.

The non-trivial direction $1 \Rightarrow 2$**.** Let R be polytime verifiable, let $c > 1$ be such that $(x, y) \in R \Rightarrow |y| \leq |x|^c$, and let D_R be the polytime machine deciding R.[1] We show an algorithm solving the search problem R.

Definition 2. *For $x \in \{0,1\}^n$, we say that $w_{pre} \in \{0,1\}^{\leq n^c}$ is a* solution prefix for *x if there is $w_{suf} \in \{0,1\}^{\leq n^c - |w_{pre}|}$ such that $(x, w_{pre} w_{suf}) \in R$.*

Fact 2.1. *If $\mathcal{P} = \mathcal{NP}$, there is a polytime machine M_{pre} deciding the language*

$$PRE = \left\{ (x, w_{pre}) : w_{pre} \text{ is a solution prefix for } x \right\} .$$

Proof. Note that $PRE \in \mathcal{NP}$, using a verifier V that gets input x, w_{pre}, expects witness w_{suf}, and outputs $D_R(x, w_{pre} w_{suf})$.[2] Since $\mathcal{P} = \mathcal{NP}$, we have that $PRE \in \mathcal{P}$. □

The algorithm solving R:

Given input $x \in \{0,1\}^n$, let w_{pre} be the empty string.

Run $M_{pre}(x, w_{pre})$, and if M_{pre} rejects then output $\perp$.

For $i = 1, ..., n^c$:

- For $b \in \{0,1\}$, run $D_R(x, w_{pre} b)$, if it accepts output $w_{pre} b$.
- Run $M_{pre}(x, w_{pre} 0)$. If it accepts extend w_{pre} by the bit 0, otherwise extend w_{pre} by the bit 1.

Output w_{pre}.

This algorithm works in n^c iterations plus a preliminary step of running M_{pre}, and in each of them it runs polytime machines. So the overall runtime is polynomial.

Let's prove correctness. If there is no y such that $(x, y) \in R$ then $x \notin L_R$, hence M_{pre} rejects in the preliminary step and the algorithm outputs $\perp$. Otherwise, M_L accepts and the algorithm enters the loop.

[1] That is, D_R gets input (x, y), runs in polytime, and accepts iff $(x, y) \in R$.

[2] If this is not obvious to you, please make sure you complete this as an exercise; it follows immediately from the definitions of $\mathcal{NP}$.

Fact 2.2. *Let $t \leq n^c$ be the iteration after which the algorithm halts. Then, for each $i \leq t$, after iteration i, the algorithm either outputs w such that $(x, w) \in R$, or has a solution prefix $w_{pre} \in \{0,1\}^i$ for x.*

Proof. We prove the invariant by induction. The base case $i = 0$ holds because when entering the loop, the algorithm has a solution prefix w_{pre} that is the empty string.

Assuming the invariant is true for iteration i, let's prove that it's true for iteration $i+1$. Let w_{pre} be the i-bit prefix entering into iteration $i+1$.

If the algorithm outputs some w then it happened in the first bullet; in this case $w = w_{pre}b$ for some $b \in \{0,1\}$, and $M_R(x, w) = 1$, hence $(x, w) \in R$. Otherwise, the algorithm concludes the iteration with prefix $w_{pre}b$ for $b \in \{0,1\}$.

- If there is w_{suf} of length $\leq n^c - (i+1)$ such that $(x, w_{pre}0w_{suf}) \in R$, then M_{pre} accepts and the algorithm continues with $w_{pre}0$, satisfying the invariant.
- Otherwise, M_{pre} rejects and the algorithm continues with $w_{pre}1$. By the induction hypothesis, there is some $w'_{suf} \in \{0,1\}^{\leq n^c - i}$ such that $(x, w_{pre}w'_{suf}) \in R$, and that w'_{suf} begins with 1 (since M_R and M_{pre} rejected). Thus, there is $w_{suf} \in \{0,1\}^{n^c-(i+1)}$ such that $(x, w_{pre}1w_{suf}) \in R$ and the invariant is satisfied. □

The algorithm halts after $t < n^c$ iterations only if outputs a string w, and by the fact above, in this case $(x, w) \in R$. Otherwise, after $t = n^c$ iterations the algorithm obtains a solution prefix w_{pre} of length n^c. Since any solution is of length at most n^c, we deduce that w_{pre} itself is a solution,[3] i.e. $(x, w_{pre}) \in R$. ■

[3] That is, w_{pre} can be extended to a solution by a suffix of length 0, hence w_{pre} is a solution