8.3 on comparison sorts).

So, to get poly time, we would need to trim off exponentially-sized chunks many times (remember problem which one could trim enough of at a high enough point that the overall algorithm could run in poly time. Not done an exhaustive search in this way. One might hope for something like a trimmed depth-first search in all the possibilities. This is certainly naive. Also not very good. In practice, however, one might not be consider SAT, for example. This one is easy. Given an expression in variables, then clearly one can try

I mean simply that the "obvious" approach will run in time exponential in the size of the input.

In general, we look at problems that have an exponential-time algorithm. We are showing is that they are unlikely to have any better solution. By "naive" exponential-time algorithm we are showing in that they have no exponential-time algorithm when viewed naively, and what


work, because we can show that $3$-CNF SAT is not more than polynomially different from ordinary SAT.

be made on other variables by squeezing the options in one of the three-term expressions. Again, this can't

an algorithm could exist that would get to poly time by having one choice of variables force the choices to
together, then the input size is $n \cdot m$. The simple assignment is still exponential time. One might think that

Consider, then, the simpler problem of $3$-CNF SAT. If we have $n$ variables and $m$ expressions, conjoined
I had given an earlier picture of how to prove basic problems \( \mathcal{NP} \)-complete:
A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number

A poly time algorithm exists even for the more realistic version of the problem in which each man and each woman is permitted to rank the acceptable spouses and the goal is to pair each individual with the number
Carey and Johnson prove
\[ \bigcap_{\exists n \in \mathbb{N}} = \cap_{\forall n \in \mathbb{N}} \]

Let \( \forall V \sqcap \forall V = V \) such that there exists a \textbf{partition} of \( V \) into \( V \) and \( \exists n \in \mathbb{N} \) for all \( n \).  

Carey and Johnson also make use of an \( N \times p \)-complete numerical problem, the partition problem.

In this latter version, this algorithm is actually quite heavily used at job fairs, in which potential employers and employees list the counterparts with whom they would like interviews and compete for a limited number of interviews slots.
Solving that then solves the arbitrary 3DM problem.

Let's start with an arbitrary instance of 3DM. This means we have coordinate sets and a collection of 3-vectors.

Proof. (Using a Restriction Argument.) Let's assume we could solve an arbitrary instance of 3XC. Then

Theorem 15.1. Exact cover by 3-sets is Np-complete.

Exactly one set C E C with exactly one set C $\subseteq S$ such that every member of $S$ occurs in that each C has cardinality 3, does there exist a subcollection $C \subseteq C$ such that every member of $S$ occurs in $S$ $\subseteq C$ such

Exact Cover by 3-Sets: Given a set S with cardinality 3g and a collection C of subsets of a given problem so that it becomes a special case of another

We can show Np-completeness by restricting a given problem so that it becomes a special case of another

15.1.1 Restriction

15.1 Methods for Proving Np-Completeness
Theorem 15.3. The subgraph isomorphism problem is $NP$-complete.

Given two graphs $G_1$ and $G_2$, does $G_1$ contain a subgraph that is isomorphic to $G_2$?

Proof: Consider the case in which $G_2$ is a clique. Then the subgraph isomorphism problem is the clique problem, any poly time solution to subgraph isomorphism would then solve the clique problem as well.

Proof: We restrict the following way. Among all minimum cover problems, consider those for which

Theorem 15.2. The minimum cover problem is $NP$-complete.

Given a set $S$, a collection $\mathcal{C} = \mathcal{C}$.

cover of size at most $S$.

such that $|\mathcal{C}|$.

and an integer $I$, does there exist a collection $\mathcal{C}$.

\begin{align*}
\mathcal{C} = \bigcup_{i \in \mathcal{C}} \mathcal{C}_i
\end{align*}
\[ \forall \alpha \in \mathbb{Z}^+ \sum_{1}^{\mathbb{Z}^+} \left( \alpha \right) = D \quad \text{if and only if} \quad \forall \alpha \in \mathbb{Z}^+ \sum_{1}^{\mathbb{Z}^+} \left( \alpha \right) = D \quad \text{where} \quad D \geq \max_{1 \leq i \leq m} \left( \alpha \right). \]

Theorem 15.5. The multi-processor scheduling is \( \text{NP-complete}. \)

Theorem 15.4. The knapsack problem is \( \text{NP-complete}. \)

Knapsack: Let \( \Omega \) be a set, \( \mathbb{Z}^+ \) be a set of positive integers, and \( \mathbb{Z}^+ \) be a set. Given a cost constraint \( \mathbb{Z}^+ \) and a value goal \( \mathbb{Z}^+ \in \mathbb{Z}^+ \), we can find a subset \( \mathbb{Z}^+ \subseteq \mathbb{Z}^+ \) of \( \mathbb{Z}^+ \) that maximizes the sum of the values, subject to the constraint that the sum of the costs is less than or equal to the constraint.
\[(\lambda) \chi \geq (t) \psi + (t) o \quad \text{or} \quad (t) \chi \geq (\lambda) \psi + (t) o \quad \text{if } \lambda \neq \lambda, \quad t \in T, \quad \text{we have}
\]

\[(t) p \geq (t) \psi + (t) o \quad \text{if}
\]

\[(t) r \leq (t) \omega \quad \text{if}
\]

A feasible schedule is a function \( d(t) \). We have a set \( L \) of tasks. For each \( t \in T \), there is a release time \( r(t) \), an execution time \( e(t) \), and a deadline.

Formally:

Sequence Within Intervals: Initially, we have a set of tasks. Each task cannot start until a certain time, and it must finish before a certain time. Is there a feasible schedule on a multiprocessor system?

The basic units individually is relatively straightforward. Consider, for example, the SAT to 3-CNF SAT.

The idea behind local replacement is that a given problem consists of a number of basic units, and transforming the 15.1.2 Local Replacement
This is the solution of the partition problem. The local replacement idea is to deal with the individual tasks.

\[
\forall a \in B, z/B = (n)^s \quad \text{and} \quad z/B = (n)^s
\]

That is, we have to be able to find \( A \setminus A' \) so that

\[
T \text{ must execute so as to fill these blocks of time exactly.}
\]

But now look at the intervals from 0 to \( z/B \) and from \( z/B + 1 \) to \( B \). By the definition of \( B' \), the tasks in

\[
\forall \tau \in B, \quad z/B = (\ell)^p \quad \text{and} \quad z/B = (\ell)^p
\]

we would have \( T \) and \( T' \). Now, since \( B \) is even, we must have \( \ell \). Since \( \ell = 2 \), it requires that \( B \) be even, since otherwise the task \( T \) imposes two conditions on feasible schedules. First, it requires that \( \tau \) and execution time \( I \).

We also construct an "enhanced" task \( T \) with \( \ell \) and execution time \( I \).

For each \( a \), we construct a task \( T^{\ell} \) with start time \( \tau \) and deadline

\[
T^{\ell} = (n)^s \quad \text{and} \quad \tau = B
\]

with size function \( s(a) \) and with \( B \) is a set A.

Proof: We're going to transform partition into this problem. Start with a partition problem with a set A.

Theorem 15.6. Segmenting within intervals is NP-complete.
correspond to triples from the original 3DM problem. The claim now is that these tests exist if and only if 

Further, in order to distinguish the original elements from each other, the remaining tests must 

of contain m and x, and thus these are the only tests that distinguish m and x from y0.
this test problem, then the collection must contain m and x, and two tests other members
Now, the new elements m and x, x, 0, y0 force the solution to have certain characteristics. If we have a solution to

\[ \exists \beta + b = r \]

\[ \{0, x\} \cap X \cap \{0, m\} \cap \{\mathcal{W} \in (h^i, x, m) : \{h^i, x, m\}\} = \mathcal{C} \]

\[ \{0, h^i, x, 0, m\} \cap X \cap X = \emptyset \]

That is, \( \{0, x\} \cap X \cap \{0, m\} \cap M \) and two tests \( \mathcal{C} \) not in m and x, we create a set \( \mathcal{W} \) \( \in (h^i, x, m) \) for every ordered triple \( (h^i, x, m) \). We create a set \( \mathcal{W} \) \( \mathcal{W} \subseteq \mathcal{W} \)

Proof. Start with a 3DM problem with a solution \( \exists \beta + b = r \). The minimum test collection problem is \( \mathcal{N}\text{-}P \)-complete.

**Theorem 15.7.** The minimum test collection problem is \( \mathcal{N}\text{-}P \)-complete.

\( \beta \in \mathcal{C} \cup \{f, a\} \) is exactly one of \( \beta \) or \( a \) for every pair \( a, b \) or \( b, c \). A, there is some test \( \mathcal{C} \in \mathcal{C} \) for which the set \( \mathcal{C} \) is of possible diagnoses from 

Can we find a subcollection \( \mathcal{C} \in \mathcal{C} \) with |\( \mathcal{C} \) such that, for every pair \( a, b \) or \( b, c \) of possible diagnoses from 

A, representable possible "tests" and a positive integer \( \beta \).

**Minimum Test Collection.** We have a finite set \( \mathcal{C} \) of subsets of
**Multiprocessor Scheduling:**

- The “coordinates,” so that a correct set of tests must correspond to a solution to the matching problem.

- By our choice of sets, there can be no overlap in every element in some test exactly once. But by our choice of sets, we have no overlap in every element of any test, and $b$ elements to test.