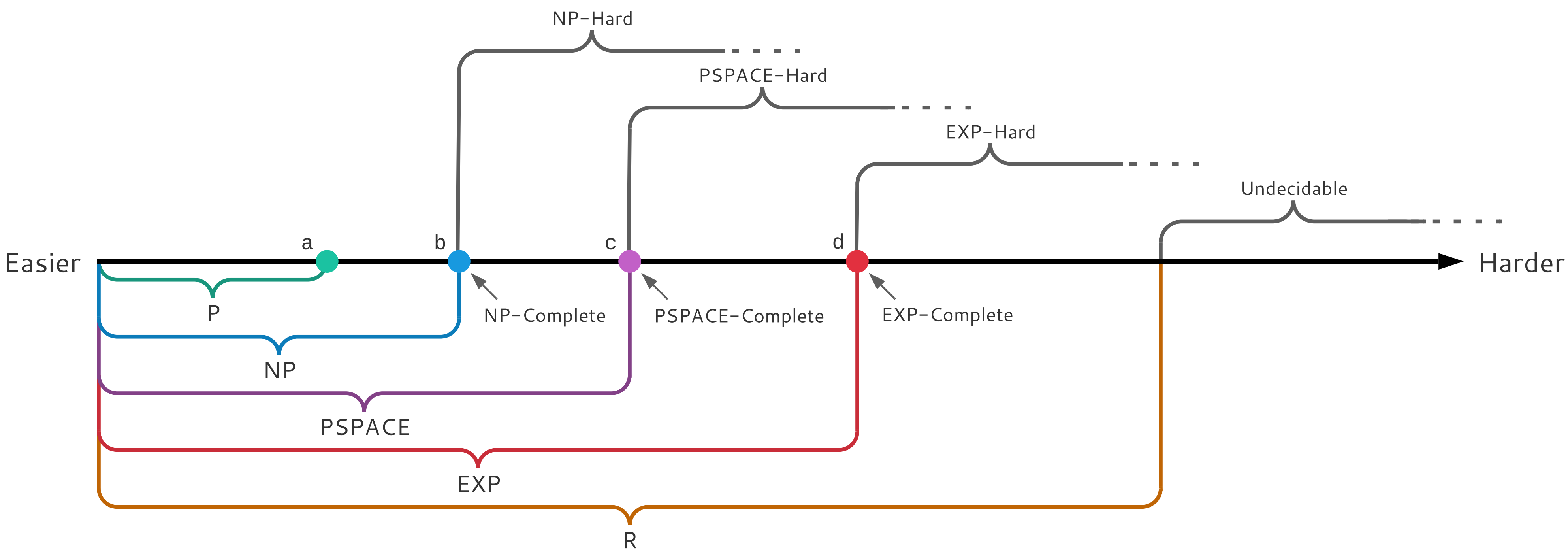
Important Computational Complexity Classes

v3 Made Oct. 12, 2021 by @discretegames, based on MIT 6.890 Lecture 1 of Fall 2014 with Erik Demaine: youtu.be/7d73E1DiH0wv



Polynomial Time
P = {problems solvable in polynomial time $O(N^k)$}

Non-Deterministic Polynomial Time
NP = {problems verifiable in polynomial time}

Polynomial Space
PSPACE = {problems solvable in polynomial space}

Exponential Time
EXP = {problems solvable in exponential time $O(2^{N^k})$}

Recursive Languages
R = {problems solvable by a Turing machine in finite time}

Hardness
X-Hard = {problems as hard as every problem in X}

Completeness
X-Complete = {problems in X and X-Hard}

By the time hierarchy theorem we know that P is strictly contained within EXP, or, viewing the diagram as a number line, that a < d.

Many assume that a < b and b < c and c < d but we aren't certain about any of those. We only know that a ≤ b ≤ c ≤ d and a < d. It could be the case that a = b = c, and thus P = NP = PSPACE.

If you can show that either a = b or (more likely) a ≠ b then you will have solved the biggest unsolved problem in computer science and the Clay Institute will award you $1,000,000. You will have solved P vs. NP.

A perhaps discouraging fact is that most problems are not even in R, that is, most problems are uncomputable.

This is because a decision problem maps all inputs to a yes or no, resulting in a powerset over the inputs. By Cantor's theorem that will have greater cardinality than the set of all possible algorithms because an algorithm is just a program, i.e. string, which can be encoded as a natural number.

Thankfully, most of the problems we care about are within EXP or even P.

There are infinite hierarchies of complexity classes not shown here, e.g. 2EXP, 3EXP, . . . Indeed there's a whole complexity zoo: complexityzoo.net