

Portfolio Question 8.1.P - Complexity Classes

Daniel Lawson University of Bristol

Portfolio 8.1.P (v4.0.0)

Portfolio Question 8.1.P - Complexity Classes

- ▶ What is a complexity class, and how does it relate to the broader topic of “algorithmic complexity”?
 - ▶ What does $P \neq NP$ mean and is it important?
 - ▶ How does complexity relate to security and data science?
 - ▶ How does complexity relate to quantum computing?

Complexity Classes

- ▶ We often do not care about the details of a certain function
- ▶ We instead ask, “Is this function in a certain complexity class?”

Polynomial Time: P

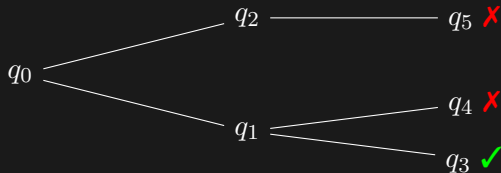
- ▶ An algorithm with time complexity $T(n)$ runs in **Polynomial Time** if $T(n) \in \cup_{i=1}^{\infty} \mathcal{O}(n^i)$.
- ▶ A language $L \in \mathbf{P}$ if there exists a Turing machine M (see lecture) such that:
 - ▶ M runs in polynomial time for all inputs
 - ▶ $\forall x \in L : M(x) = 1$
 - ▶ $\forall x \notin L : M(x) = 0$

Examples of algorithms in P

- ▶ **Primality Testing**: is a number x a prime number?
- ▶ **Shortest Path** in a graph: given two nodes, what is the shortest path? (for example, **Dijkstra's Algorithm**)
- ▶ **Minimal Weighted Matching**: Given n jobs on n machines with cost matrix c_{ij} , how do we allocate jobs? Solvable as an **integer program**.
- ▶ **Pattern Matching**: Asking, is a given **pattern present** in the data? The runtime depends on the data structure and pattern, but broad classes are solvable (e.g. **graphs**)

Non-Determinism

- ▶ A **Non-Deterministic** Turing machine is like a Turing Machine, except δ can go to multiple states for the same input.
- ▶ When a choice of transition is given, the Non-Deterministic Turing Machine “takes them all simultaneously”.
- ▶ The machine accepts if any of the paths accept.



Non-Deterministic Polynomial Time: NP

- ▶ A language $L \in \text{NP}$ if there exists a **Non-Deterministic** Turing machine M such that:
 - ▶ M runs in **Polynomial Time** for all inputs
 - ▶ $\forall x \in L : M(x) = 1$
 - ▶ $\forall x \notin L : M(x) = 0$

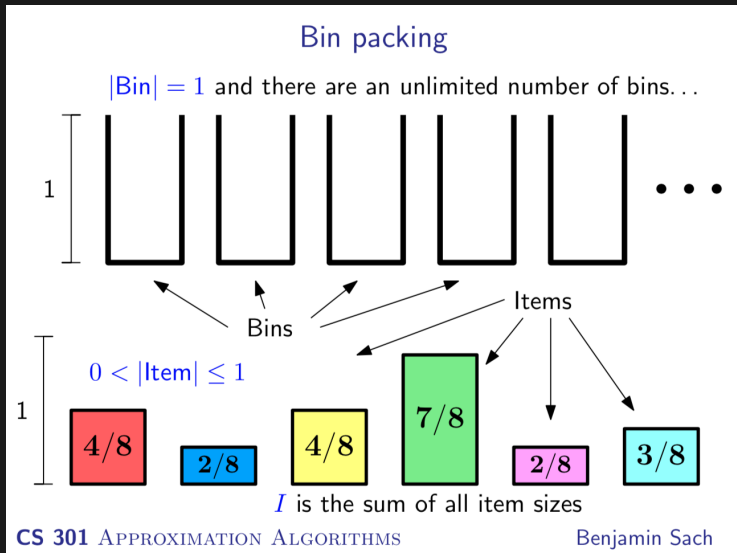
Examples of algorithms in NP

- ▶ **Travelling salesman problem**: Given a distance matrix between n cities, is there a route between them all with total distance less than D ?
- ▶ **Bin packing**: Can you place n items into as few fixed-size bins as possible?
- ▶ **Boolean satisfiability**: Is a set of boolean logic statements true?
- ▶ **Integer factorisation**: Given a number x , what are its primes?

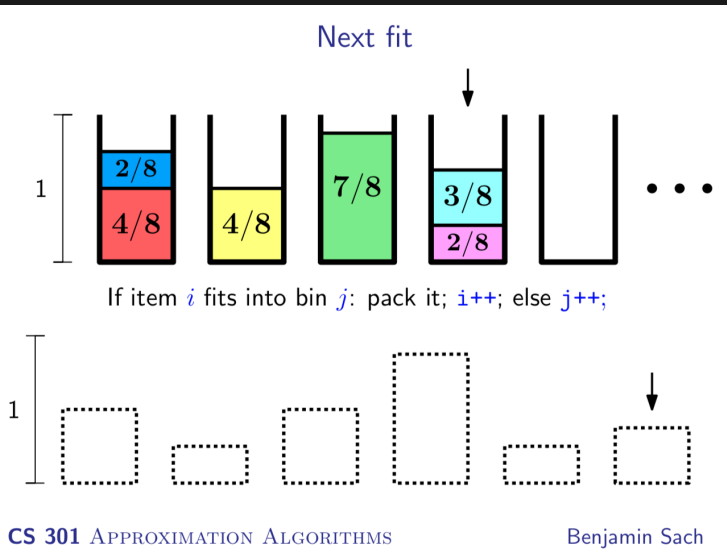
Data science consequences

- ▶ Having an algorithm is the easiest way to prove that f is in a complexity class.
 - ▶ It is hard to prove that a problem is not in P!
- ▶ Many exact problems seem to be NP.
- ▶ We can sometimes do very well with an **approximate algorithm** in P. Examples:
 - ▶ Travelling salesman: exactly solved for Euclidean distances, Christofides and Serdyukov's approximation using minimum weight perfect matching
 - ▶ Bin packing. . .
- ▶ Quantifying approximation error is therefore very important!

Bin packing problem

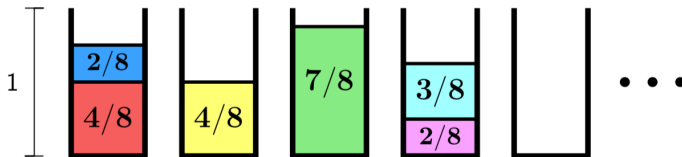


Bin packing: next fit



Bin packing: next fit

Next fit



Next fit runs in $O(n)$ time but how good is it?

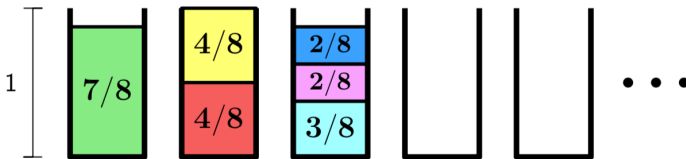
- Let $\text{fill}(i)$ be the sum of item sizes in bin i
and b the number of non-empty bins (using Next fit)
- Observe that $\text{fill}(2i-1) + \text{fill}(2i) > 1$ (for $1 \leq 2i \leq b$)

$$\text{so } \lfloor b/2 \rfloor < \sum_{1 \leq 2i \leq b} \text{fill}(2i-1) + \text{fill}(2i) \leq I \leq \text{Opt}$$

Next fit is an 2-approximation for bin packing which runs in linear time

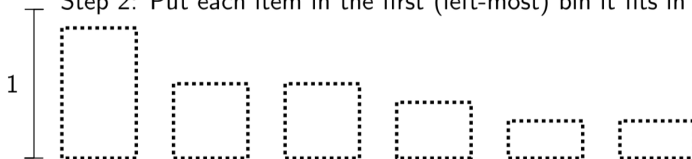
Bin packing: first fit decreasing

First fit decreasing (FFD)



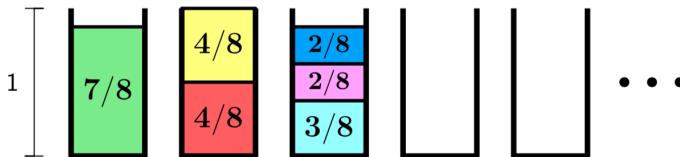
Step 1: Sort the items into non-increasing order

Step 2: Put each item in the first (left-most) bin it fits in



Bin packing: first fit decreasing

First fit decreasing (FFD)



- Consider bin $j = \lceil \frac{2b}{3} \rceil$ (FFD uses b bins on this instance)

Case 2: Bin j contains only items of size $\leq 1/2$

As $\lceil 2k/3 \rceil - 1 < I$

we have that $\lceil 2k/3 \rceil - 1 \leq 2k/3 \leq \text{Opt}$

- So FFD is a $3/2$ -approximation for bin packing

Addendum

- ▶ Complexity classes are not everything!
- ▶ Some examples of algorithms in P^1 :
 - ▶ Max-Bisection is approximable to within a factor of 0.8776 in around $O(n^{10^{100}})$ time
 - ▶ Energy-driven linkage unfolding algorithm is at most $117607251220365312000n^{79}(l_{max}/d_{min}(\Theta_0))^{26}$
 - ▶ The classic “picture dropping problem” for how to wrap string such that it that will drop when one nail is removed, with n nails, can be solved in $O(n^{43737})$
 - ▶ Approximate algorithms (accurate to within $(1 + \epsilon)$ often scale badly, e.g. $O(n^{1/\epsilon})$

¹Stack Exchange **Polynomial Time algorithms with huge exponent**

Discussion

- ▶ Can you relate quantum complexity classes to the discussion above?
- ▶ What do complexity classes say about whether a problem is “solveable”?
- ▶ What is the complexity of some machine learning algorithms, e.g. Neural Networks?
- ▶ Can you find examples of encryption algorithms that relate to complexity?
 - ▶ How about cryptocurrency?