Algorithms II Cheat Sheet

Tips

Apply an algorithm you know in a clever way, don't write a new algorithm.

Set notation

$A \in [10] \equiv A \in [1..10]$

Big O

Notation	Intuitive meaning	Analogue
$f(n) \in O(g(n))$	f grows at most as fast as g	$\leq$
$f(n) \in \Omega(g(n))$	f grows at least as fast as g	$\geq$
$f(n) \in \Theta(g(n))$	f at the same rate as g	$=$
$f(n) \in o(g(n))$	f grows strictly less fast than g	$<$
$f(n) \in \omega(g(n))$	f grows strictly faster than g	$>$

Notation	Formal definition
$f(n) \in O(g(n))$	$\exists C,\, n_0 \colon \forall n \geq n_0 \colon f(n) \leq C \cdot g(n)$
$f(n) \in \Omega(g(n))$	$\exists c,\, n_0 \colon \forall n \geq n_0 \colon f(n) \geq c \cdot g(n)$
$f(n) \in \Theta(g(n))$	$\exists c,\, C,\, n_0 \colon \forall n \geq n_0 \colon c \cdot g(n) \leq f(n) \leq C \cdot g(n)$
$f(n) \in o(g(n))$	$\forall C \colon \exists n_0 \colon \forall n \geq n_0 \colon f(n) \leq C \cdot g(n)$
$f(n) \in \omega(g(n))$	$\forall c \colon \exists n_0 \colon \forall n \geq n_0 \colon f(n) \geq c \cdot g(n)$

Interval Scheduling

A **request** is a pair of integers (s, f) with $0 \leq s \leq f$.
We call s the **start time** and f the **finish time**.

A set A of requests is **compatible** if for all distinct $(s, f), (s', f') \in A$, either $s' \geq f$ or $s \geq f'$ — that is, the requests' time intervals don't overlap.

Interval Scheduling Problem
Input: An array $\mathcal{R}$ of n requests $(s_1, f_1), \ldots, (s_n, f_n)$.
Desired Output: A compatible subset of $\mathcal{R}$ of maximum possible size.

Algorithm: GreedySchedule
Input: An array $\mathcal{R}$ of n requests.
Output: A maximum compatible subset of $\mathcal{R}$.

```
1 begin
2     Sort $\mathcal{R}$'s entries so that $\mathcal{R} \leftarrow [(s_1, f_1), \ldots, (s_n, f_n)]$ where $f_1 \leq \cdots \leq f_n$.
3     Initialise $A \leftarrow []$, lastf $\leftarrow 0$.
4     foreach $i \in \{1, \ldots, n\}$ do
5         if $s_i \geq$ lastf then
6             Append $(s_i, f_i)$ to $A$ and update lastf $\leftarrow f_i$.
7     Return $A$.
```

Complexity:
Step 2 takes O(n log n)
Steps 3–6 all take O(1) time and are executed at most n times.
$\therefore\ totalrunningtime = O(nlogn) + O(n)O(1) = O(nlogn)$.

Interval Scheduling

Formal GreedySchedule
$A^+ :=$ argmin $\{f : (s, f) \in R, A \cup \{(s, f)\}$ is compatible$\}$ for all $A \subseteq R$,
$A_0 := \emptyset, \qquad A_{i+1} := A_i \cup \{A_i^+\}$
t := max{i: A_i is defined}

Interval Scheduling Proofs
Lemma: Greedy Schedule always outputs A_t
Proof: By induction form the following loop invariant. At the start of the i'th iteration of 4-7:

- A is equal to $A_t \cap \{(s_1, f1), ..., (s_{i-1}, f_{i-1})\}$
- lastf is equal to the latest finish time of any request in A (or 0 if A = [])

Lemma: A_t is a compatible set
Proof: Instant by induction; A_0 is compatible, and if A_i is compatible then so is $A_{i+1} = A_i \cup A^+$ by the definition of A_i^+

Lemma: A_t is a maximum compatible subset of the Array R (look in pseudocode)
Proof:
Base case for i = 1: A_0^+ is the fastest finishing request in R by definition
Inductive step: Suppose A_i finishes faster than B_i. Let B_i^+ be the (i+1)'st fastetst-finishing element of B. Since A_i finishes faster than B_i, $A_i \cup \{B_i^+\}$ is compatible. Hence by definition, A_i^+ exists and finishes no later than B_i^+

Theorem: GreedySchedule outputs A_t, which is a maximum compatible set.
Proof: putting all of the above proofs together, we prove the theorem.

Graph Theory

Definitions:

- Graph:
- Edge
- Vertex
- component
- degree
- walk
- isomorphism
- euler walk
- connected
- digraph
- strongly connected
- weakly connected
- in-degree
- out-degree
- cycle
- hamilton cycle
- bijection

ODEs

1st Order Linear	Use integrating factor, $I = e^{\int P(x)dx}$
Separable:	$\int P(y)dy/dx = \int Q(x)$
HomogEnEous:	$dy/dx = f(x,y) = f(xt,yt)$ sub $y = xV$ solve, then sub $V = y/x$
Exact:	If $M(x,y)+N(x,y)dy/dx = 0$ and $M_y = N_x$ i.e. $\langle M,N\rangle = \nabla F$ then $\int_x M + \int_y N = F$
Order Reduction	Let $v = dy/dx$ then check other types *If purely a function of y,* $\frac{dv}{dx} = v\frac{dv}{dy}$
Variation of Parameters:	When $y''+a_1y'+a_2y = F(x)$ F contains $\ln x$, $\sec x$, $\tan x$, $\div$
Bernoulli	$y' + P(x)y = Q(x)y^n$ $\div y^n$ $y^{-n}y' + P(x)y^{1-n} = Q(x)$ *Let* $U(x) = y^{1-n}(x)$ $\frac{dU}{dx} = (1-n)y^{-n}\frac{dy}{dx}$ $\frac{1}{1-n}\frac{du}{dx} + P(x)U(x) = Q(x)$ *solve as a 1st order*
Cauchy-Euler	$x^ny^n + a_1x^{n-1}y^{n-1} + \cdots + a_{n-1}y^{n-2} + a_ny = 0$ guess $y = x^r$
3 Cases:	
1) Distinct real roots	$y = ax^{r_1} + bx^{r_2}$
2) Repeated real roots	$y = Ax^r + y_2$ *Guess* $y_2 = x^ru(x)$ *Solve for* $u(x)$ *and choose one* $(A = 1, C = 0)$
3) Distinct complex roots	$y = B_1x^a\cos(b\ln x) + B_2x^a\sin(b\ln x)$

Laplace Transforms

$L[f](s) = \int_0^\infty e^{-sx}f(x)dx$

$f(t) = t^n, n \geq 0$	$F(s) = \frac{n!}{s^{n+1}}, s > 0$
$f(t) = e^{at}, a$ *constant*	$F(s) = \frac{1}{s-a}, s > a$
$f(t) = \sin bt, b$ *constant*	$F(s) = \frac{b}{s^2+b^2}, s > 0$
$f(t) = \cos bt, b$ *constant*	$F(s) = \frac{s}{s^2+b^2}, s > 0$
$f(t) = t^{-1/2}$	$F(s) = \frac{\pi}{s^{1/2}}, s > 0$
$f(t) = \delta(t-a)$	$F(s) = e^{-as}$
f'	$L[f'] = sL[f] - f(0)$
f''	$L[f''] = s^2L[f] - sf(0) - f'(0)$
$L[e^{at}f(t)]$	$L[f](s-a)$
$L[u_a(t)f(t-a)]$	$L[f]e^{-as}$

Vector Spaces

$v_1, v_2 \in V$

1. $v_1 + v_2 \in V$
2. $k \in \mathbb{F}, kv_1 \in V$
3. $v_1 + v_2 = v_2 + v_1$
4. $(v_1 + v_2) + v_3 = v_1 + (v_2 + v_3)$
5. $\forall v \in V, 0 \in V \mid 0 + v_1 = v_1 + 0 = v_1$
6. $\forall v \in V, \exists -v \in V \mid v + (-v) = (-v) + v = 0$
7. $\forall v \in V, 1 \in \mathbb{F} \mid 1 * v = v$
8. $\forall v \in V, k, l \in \mathbb{F}, (kl)v = k(lv)$
9. $\forall k \in \mathbb{F}, k(v_1 + v_2) = kv_1 + kv_2$
10. $\forall v \in V, k, l \in \mathbb{F}, (k + l)v = kv + lv$