APPROXIMATION ALGORITHMS

ROUNDING DATA & DYNAMIC PROGRAMMING

RASMUS PAGH
UNIVERSITY OF COPENHAGEN

TODAY:

ROUNDING PROBLEMS TO MAKE THEM FEASIBLE

CASE STUDIES:

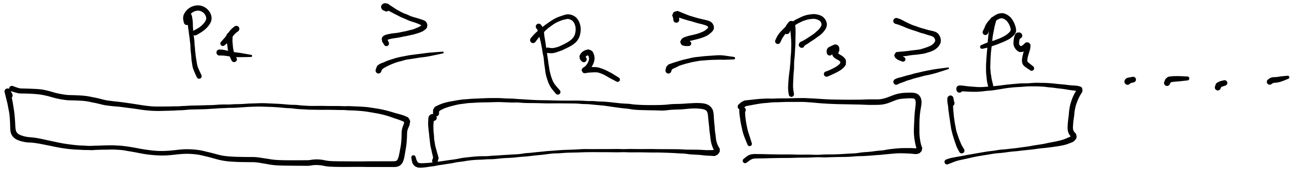
- SCHEDULING PARALLEL JOBS $(2 \cdot OPT \rightsquigarrow (1+\varepsilon) \cdot OPT)$
- BIN-PACKING

RECALL THE PROBLEM:

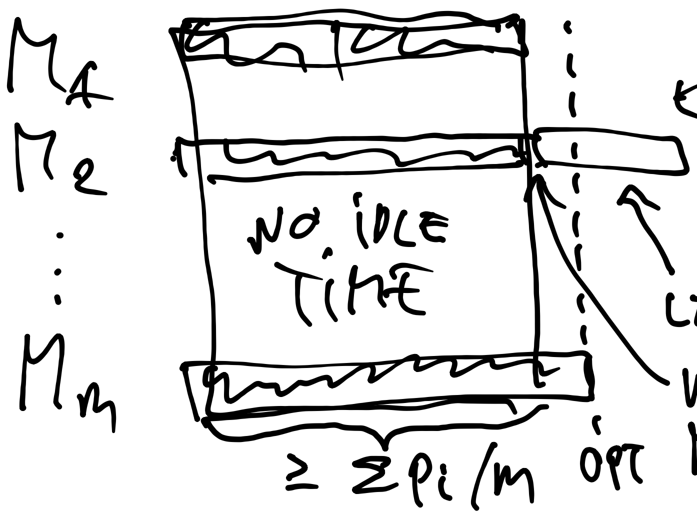
SCHEDULE JOBS OF DURATION $p_1, \dots, p_n \geq 0$ ON m MACHINES.

OPTIMIZE MAKESPAN, TOTAL TIME TO FINISH ALL JOBS.

TODAY'S APPROACH: PROCESS LONGEST JOBS FIRST.



FEASIBLE SOLUTION HAS A LAST JOB TO FINISH



IF $p_j \leq \varepsilon$ OPT, WE HAVE A $1+\varepsilon$ APPROX.

LAST JOB, LENGTH p_j

WHEN JOB j STARTED, ALL MACHINES WERE BUSY

LET $T = \sum_i p_i / m$. SPLIT JOBS: $p_i < T/k$: SHORT JOB
($k \approx 1/\varepsilon$) $p_i \geq T/k$: LONG JOB

ALGORITHMIC IDEA:

- FIRST SCHEDULE THE LONG JOBS OPTIMALLY (OR NEARLY SO)
- THEN SCHEDULE THE SHORT JOBS GREEDILY IN DECREASING ORDER.

PAUSE AND THINK:

- THERE CAN BE AT MOST mk LONG JOBS. WHY?
- HOW MANY WAYS CAN WE SCHEDULE THE LONG JOBS?

TWO CASES:

- THE LAST JOB j TO FINISH IS SHORT: COST IS $OPT + p_j \leq OPT + \frac{OPT}{k}$
- THE LAST JOB TO FININS IS LONG: COST IS SAME AS WITHOUT SHORT JOBS, SO EQUALS OPT!

PREVIOUS APPROACH GIVES RUNNING TIME $O(m^{km})$.
CAN WE DO BETTER FOR LARGE m?

REDEFINE T: UPPER BOUND ON OPT.

GOAL: FIND SCHEDULE WITH MAKESPAN $\leq (1+\frac{1}{k})T$

ROUNDING: ROUND DOWN p_i TO NEAREST MULTIPLE OF T/k^2 ← AT MOST k^2 POSSIBLE LENGTHS

FIND SCHEDULE FOR LONG JOBS USING DYNAMIC PROGRAMMING ← AT MOST k JOBS/MACHINE IN SCHEDULE OF LENGTH $\leq T$

- $OPT(n_1, n_2, \ldots, n_{k^2}) = 1 + \min OPT(n_1 - s_1, \ldots, n_{k^2} - s_{k^2})$

  ↑ SMALLEST # MACHINES NEEDED FOR WORKLOAD

  ↑ MIN OVER ALL CONFIGURATIONS OF ONE MACHINE

- COMPUTE BOTTOM-UP FOR $\sum n_i = 0, 1, \ldots, n^{k^2}$ UNTIL REACHING OPT FOR INPUT CONFIG.

HOW TO FIND T? OPTION 1: TRY $T = 1, 2, \ldots$ OPTION 2: SEE BOOK

FINAL ALGORITHM RUNS IN TIME $n^{f(k)}$

- CAN MAKE APPROXIMATION ARBITRARILY GOOD BY INCREASING k.

$\Rightarrow$ A POLYNOMIAL TIME APPROXIMATION SCHEME

BUT: TIME GROWS EXPONENTIALLY WITH $1/\varepsilon$ FOR APPROXIMATION FACTOR $1+\varepsilon$

LATER: FULLY POLYNOMIAL TIME APPROX. SCHEME: POLYNOMIAL IN n AND $1/\varepsilon$.

BIN-PACKING PROBLEM

INPUT: $a_1, a_2, a_3, \ldots, a_n \in (0,1)$

OBJECTIVE: MAP $\{1,\ldots,n\}$ TO m BINS, SUCH THAT ELEMENTS IN EACH BIN HAVE SUM ≤ 1, AND m IS SMALLEST POSSIBLE

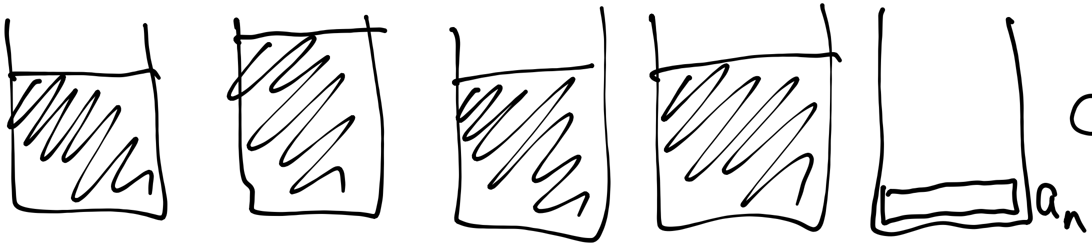
FACT: IT IS NP-HARD TO DISTINGUISH BETWEEN THE CASE WHERE OPT$=2$ AND OPT$=3$ $\Rightarrow$ $\frac{3}{2}$-APPROX IS BEST POSS.

BUT: CAN ALMOST GET AN APPROXIMATION SCHEME: FOR EVERY $\varepsilon > 0$, CAN PACK INTO $(1+\varepsilon)\mathrm{OPT} + O_\varepsilon(1)$ BINS IN POLYNOMIAL TIME.

OBSERVATION: "SMALL" ITEMS OF SIZE $a_i < \varepsilon$ DO NOT MATTER.
(WE CAN PLACE THEM LAST)

ASSUME $a_1 \geq a_2 \geq \dots \geq a_n$;

WHERE CAN WE PLACE a_n, IF $a_1, \dots, a_{n-1}$ ARE PLACED?



CASE 1: SOME BIN HAS SPARE CAPACITY ε

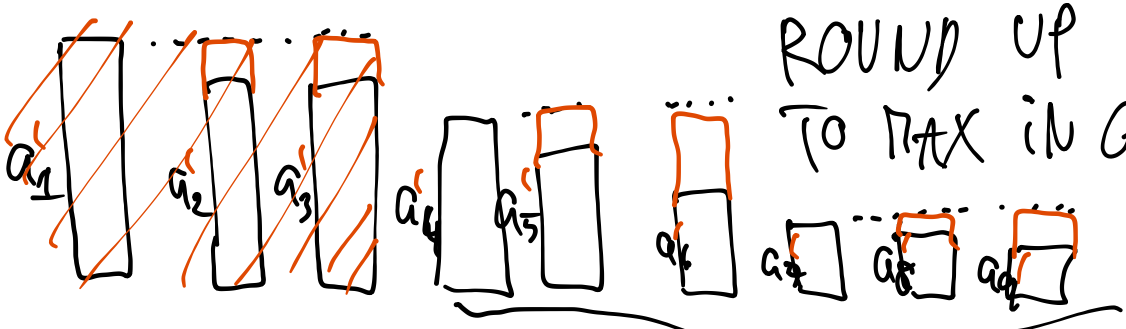
CASE 2: NO BIN HAS SPARE CAPACITY ε
→ NEED NEW BIN

CONCLUSION:
SUFFICES TO SOLVE THE PROBLEM IN THE CASE WHERE $a_1, \dots, a_n \geq \varepsilon/2$

$\boxed{\text{PAUSE AND THINK: WHY DO WE STILL HAVE A SOLUTION CLOSE TO OPT IN CASE 2?}}$

DISCRETIZATION VIA GROUPING

ROUND UP SIZE TO MAX IN GROUP



DROP LARGEST GROUP

NEW INPUT I' WITH n/k DISTINCT VALUES

⇓

CAN USE SCHEDULING ALGORITHM FROM FIRST PART TO FIND $1+\varepsilon$ APPROX.

CLAIM:

$$OPT(I') \overset{1)}{\le} OPT(I) \le OPT(I') + k$$

1) GIVEN SOLUTION TO I, CAN GET SOLUTION TO I' OF AT MOST SAME COST

2) GIVEN SOLUTION TO I', CAN GET SOLUTION TO I WITH ADDITIONAL COST $\le k$.

CHOOSE $k = \lfloor \varepsilon \sum_i a_i \rfloor$ TO GET SOLUTION OF COST $(1+\varepsilon)OPT + O_\varepsilon(1)$