# A Greedy Strategy

### List Scheduling:

Order all processes in a list. When a machine runs empty assign the next yet unprocessed job to it.

#### Alternatively:

Consider processes in some order. Assign the i-th process to the least loaded machine.

It is easy to see that the result of these greedy strategies fulfill the local optimally condition of our local search algorithm. Hence, these also give 2-approximations.

| EADS II                | 15 Scheduling on Identical Machines: Greedy |     | חר   |
|------------------------|---|-----|------|
| 🛛 💾 🛛 🖉 © Harald Räcke |   | 299 | ן טנ |

### Proof:

- Let p<sub>1</sub> ≥ · · · ≥ p<sub>n</sub> denote the processing times of a set of jobs that form a counter-example.
- Wlog. the last job to finish is n (otw. deleting this job gives another counter-example with fewer jobs).
- If p<sub>n</sub> ≤ C<sup>\*</sup><sub>max</sub>/3 the previous analysis gives us a schedule length of at most

$$C_{\max}^* + p_n \le \frac{4}{3} C_{\max}^*$$

Hence,  $p_n > C_{\max}^*/3$ .

- This means that all jobs must have a processing time  $> C_{\text{max}}^*/3$ .
- But then any machine in the optimum schedule can handle at most two jobs.
- > For such instances Longest-Processing-Time-First is optimal.

15 Scheduling on Identical Machines: Greedy

# A Greedy Strategy

#### Lemma 2

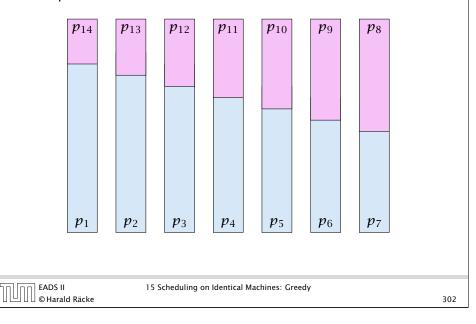
If we order the list according to non-increasing processing times the approximation guarantee of the list scheduling strategy improves to 4/3.

#### EADS II © Harald Räcke

15 Scheduling on Identical Machines: Greedy

When in an optimal solution a machine can have at most 2 jobs the optimal solution looks as follows.

300



- We can assume that one machine schedules p<sub>1</sub> and p<sub>n</sub> (the largest and smallest job).
- If not assume wlog. that p<sub>1</sub> is scheduled on machine A and p<sub>n</sub> on machine B.
- Let *p<sub>A</sub>* and *p<sub>B</sub>* be the other job scheduled on *A* and *B*, respectively.
- $p_1 + p_n \le p_1 + p_A$  and  $p_A + p_B \le p_1 + p_A$ , hence scheduling  $p_1$  and  $p_n$  on one machine and  $p_A$  and  $p_B$  on the other, cannot increase the Makespan.
- Repeat the above argument for the remaining machines.

|                        | 15 Scheduling on Identical Machines: Greedy |     |
|------------------------|---|-----|
| 🛛 💾 🛛 🖉 © Harald Räcke |   | 303 |

| L |  |  |
|---|--|--|

