Online reassignment models in ~~suchglinde~~ scheduling

(on two related machines)

György Dósa,
University of Pannonia,
Hungary

FIELDS INSTITUTE,
September 27, 2011

My scientific origin

- Msc: ELTE (Eötvös Loránd Univ.), Budapest, Hungary, 1987
- PhD: Szeged, Hungary, 2009
  - from the hands of J. Csirik,
  - supervisor: B. Vizvari,
  - basically on the common works with Yong He
- Yong He, Hangzhou, 7 papers
- Zsolt Tuza, Leah Epstein, Xin Han,…
- I work at Univ. of Pannonia, Hungary, near Balaton (biggest lake in middle Europe)

Hangzhou, (a bit left from Sanghai), and the West Lake

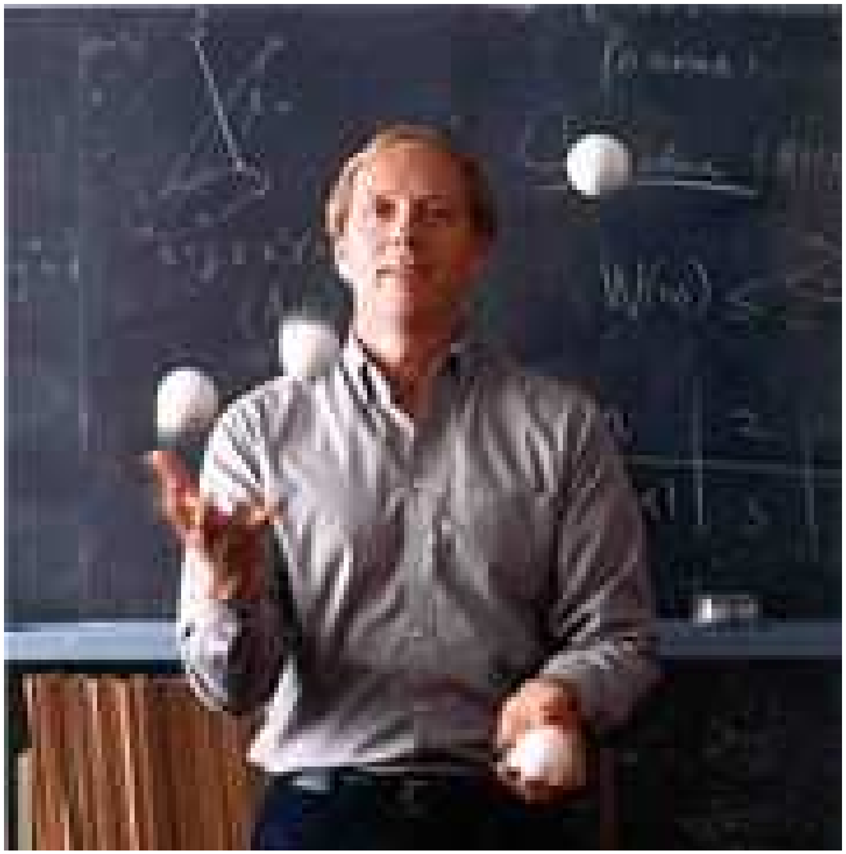
„Above is Heaven, here is Hangzhou”

My numbers:

Erdős-number=2	Dosa – Zs. Tuza – P. Erdos
Bezdek-number=3	Dosa – Tuza – - T.I.Zamfirescu – K. Bezdek
Deza-number=4	Dosa - He - Frank K. Hwang – - Samuel Onn – Antoine Deza
Lorea-number=4	Dosa – He - G. Woeginger – J. Urrutia – Jesus A. De Lorea
Mitchell-number=3	Dosa- He – G. Woeginger – - Joseph S.B. Mitchell
Ye-number=4	Dosa- He – En Yu Yao – Jie Sun – Yinyu Ye

some words about scheduling

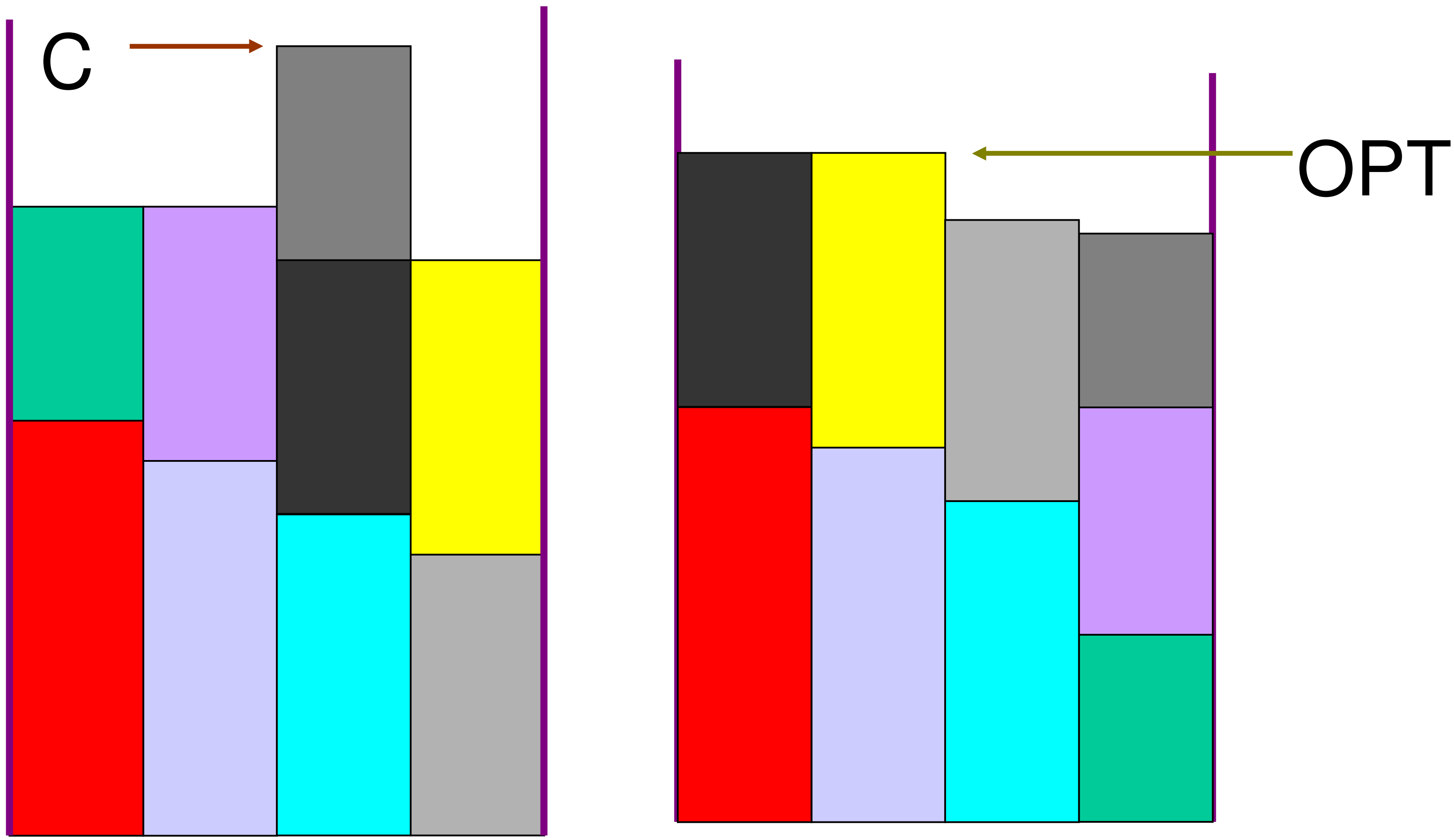
the father: Ronald Graham



with balls

with his wife and Erdős

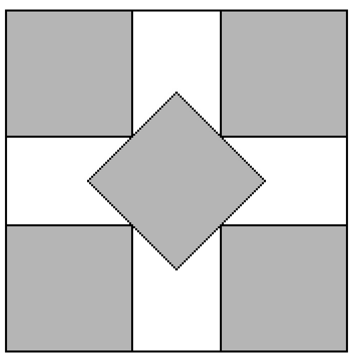
The beginning: Graham's algorithms LS (and it's ordered version, LPT), 1966



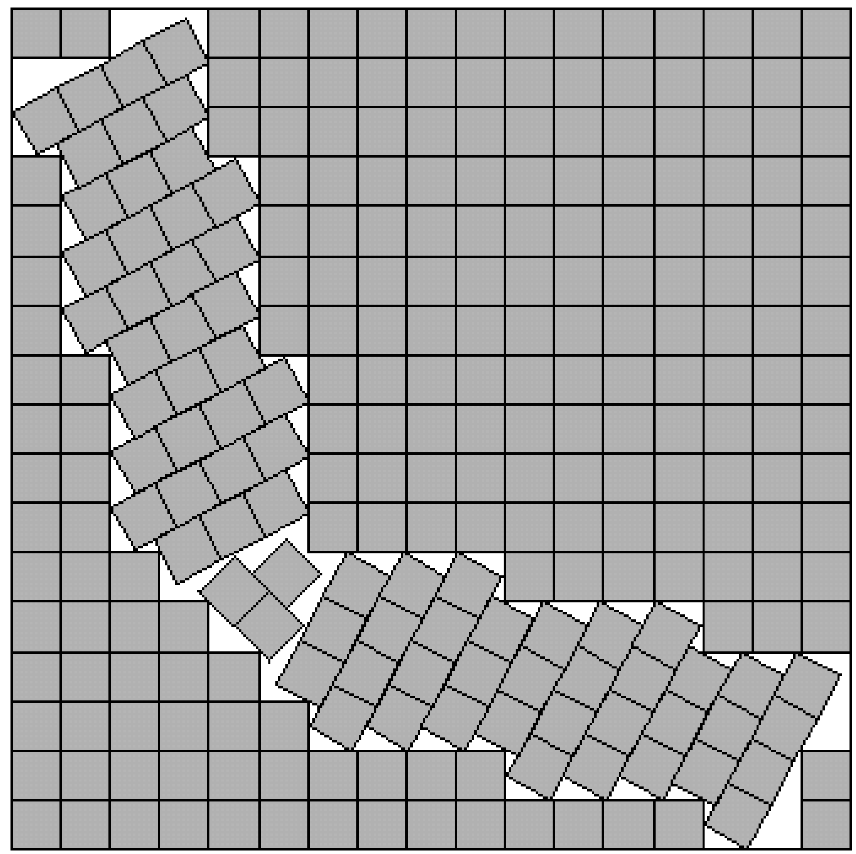
C (=maximum completion time = makespan)→ min

a 40 years old problem of Graham and Erdős: k equal squares should be packed into the smallest (big) square

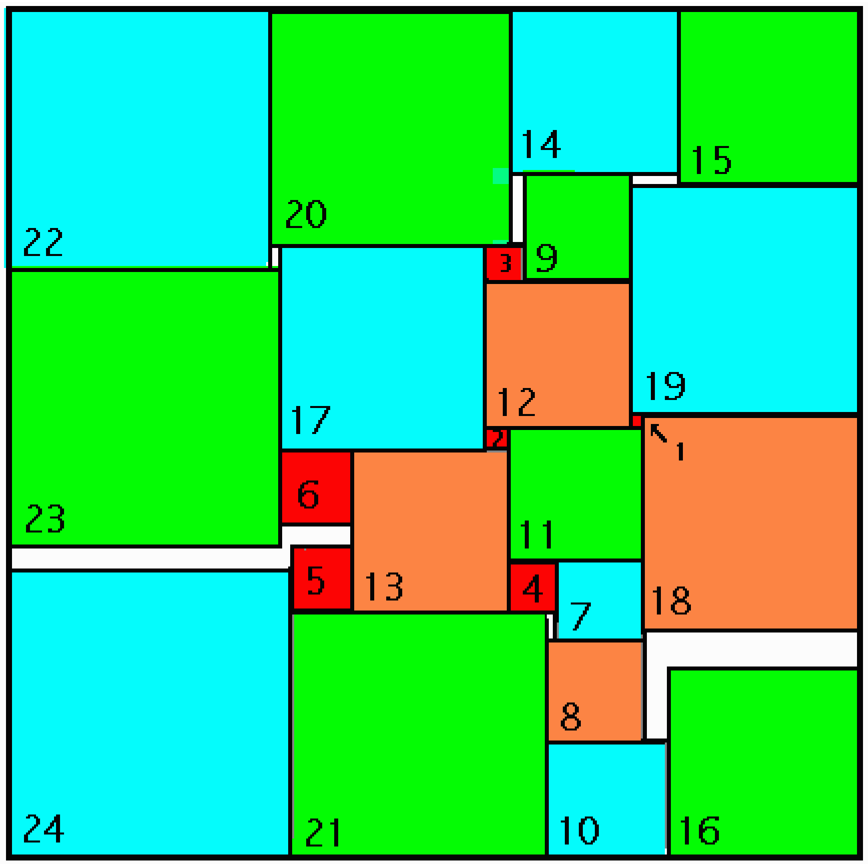
k=5



k=272, they fit into 17X17



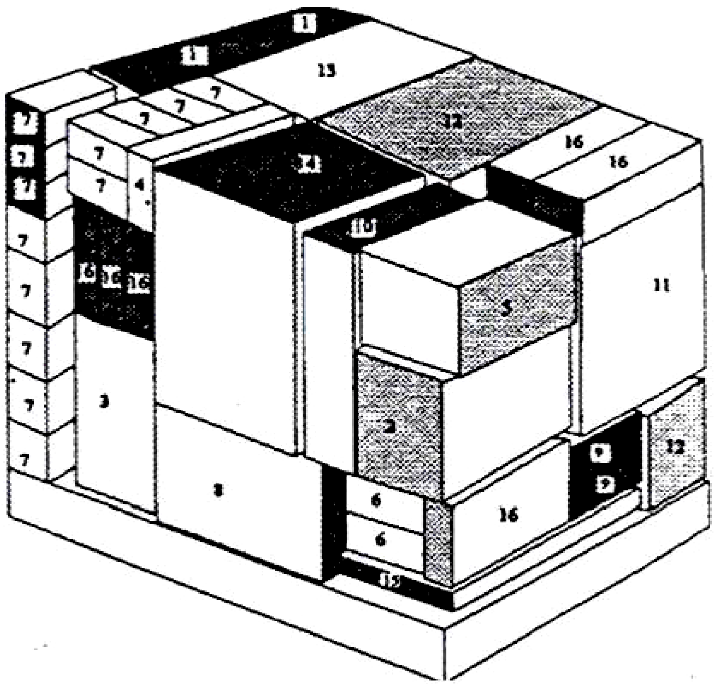
Or: there is one square with size k, for 1 $\leq$ k $\leq$ 24 (for example)



This is 71 x 71,
(by M. Hujter)

is there smaller big square?

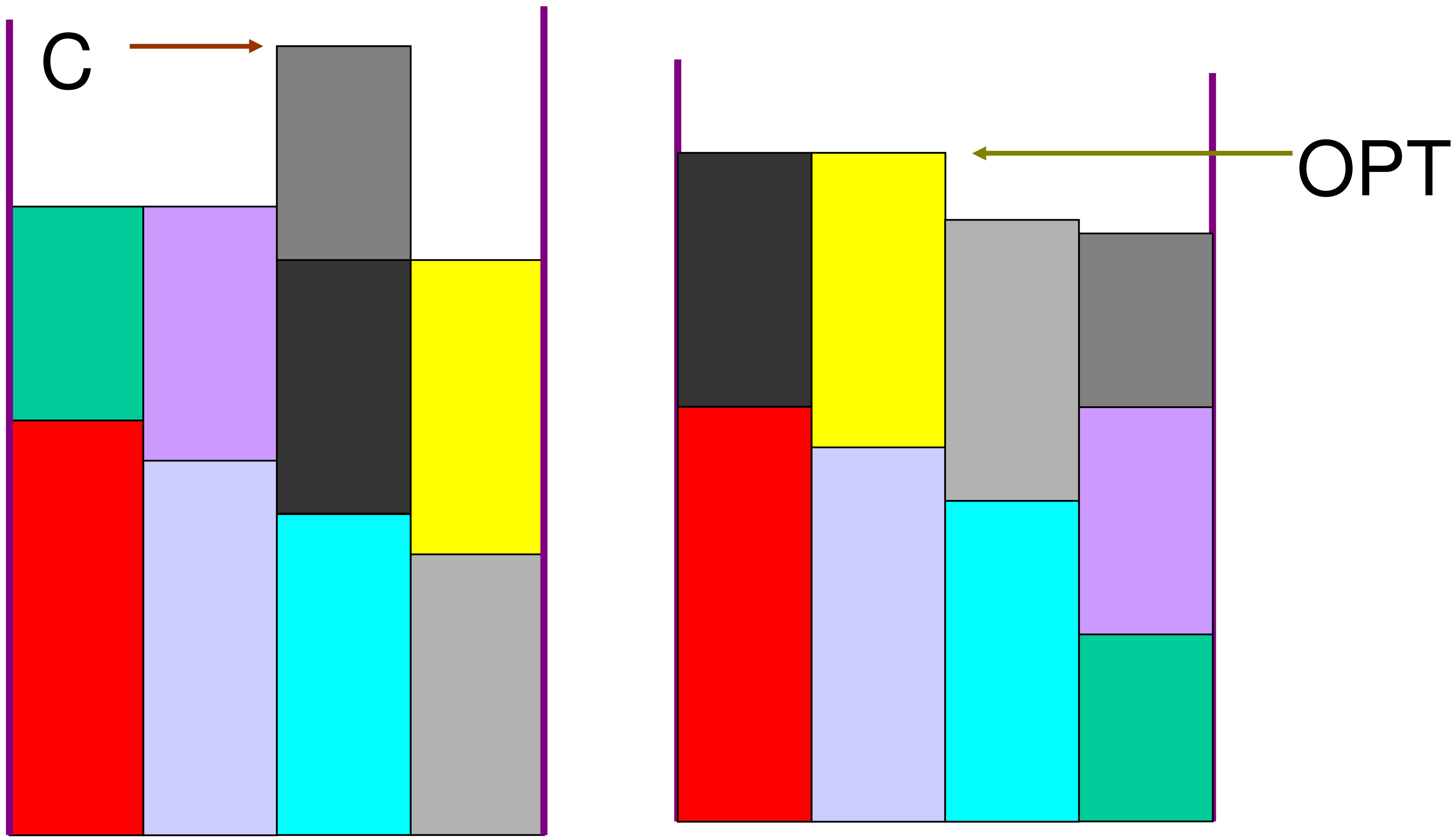
a three dimensional packing problem:



computer aided packing plan, A. Mészáros

So: scheduling (or packing) can be seen as a special kind of „discrete geometry"
(in visualization)

LS (or other algorithm) again:
How bad can be Alg/OPT?



How big (bad) can be ALG/OPT=? (approx. or competitive ratio)

- LS with decreasing sizes: $\approx$ 4/3

(LPT/OPT is not worse, not bigger than $\approx$ 4/3)

- LS generally: not worse than 2-1/m $\approx$ 2

offline: we know everything about the input

online: we know nothing in advance (but we must make decisions without knowledge of the future)

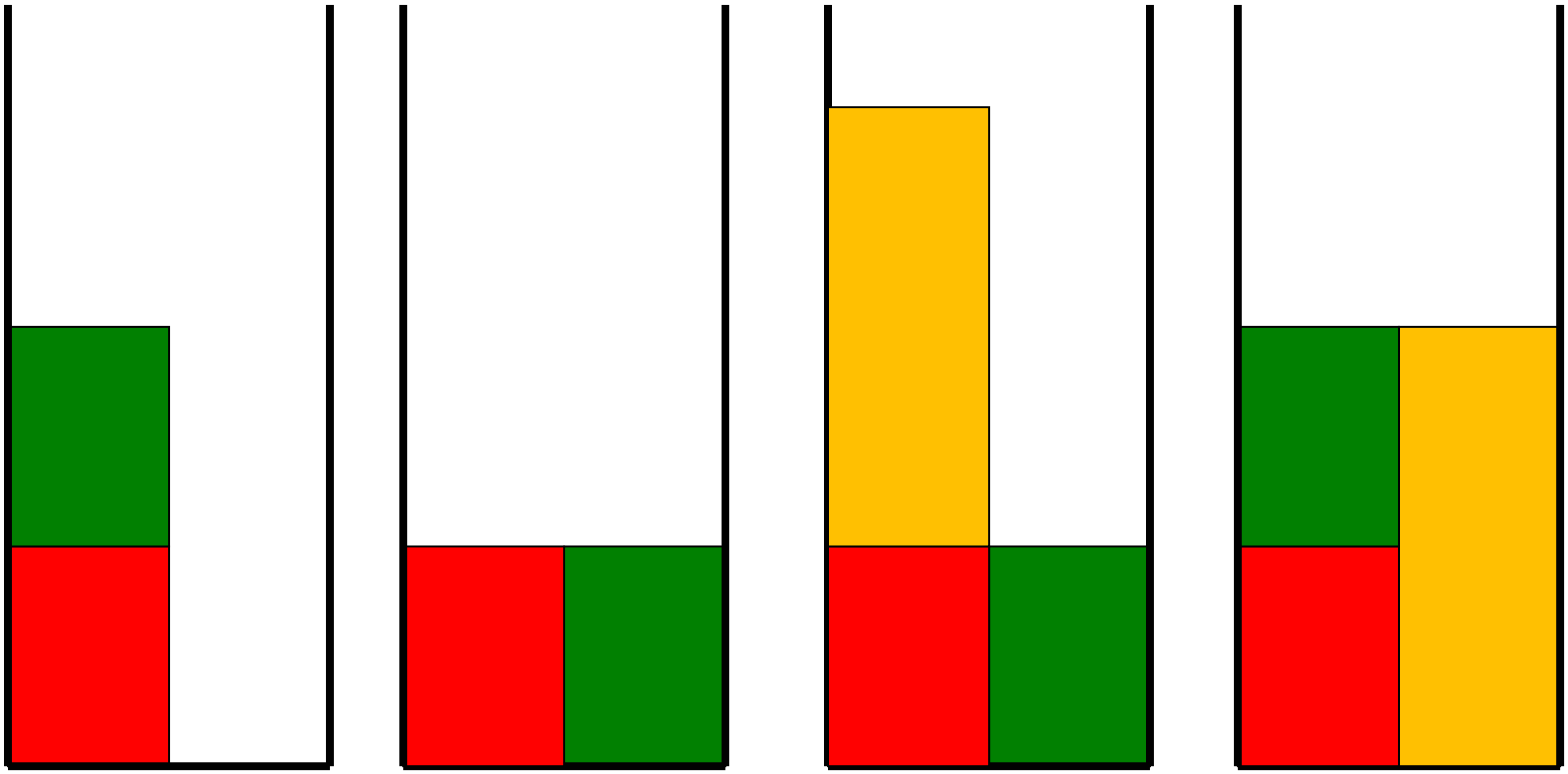
semi online: we know some things, but not everything (between offline and online)

(decreasing sizes helps to make ALG/OPT smaller)!

offline scheduling (of parallel machines)

- getting optimal solution needs exponential many time (NP-hardness)
- solution within (1+ϵ) x OPT can be got in pol. (but still much) time, (approx. scheme)
- there are fast heuristics

online scheduling: lower bounds (by adversaries) and optimality



Thus C/OPT can not be better (always) than 3/2

online scheduling (of parallel machines)

- Graham's algorithm is optimal for two machines, (C/OPT is at most 3/2)
- LS is also optimal for m=3, (Faigle, Kern, Turán, 1989)
- Four machines (m=4) ?
- We do not know optimal algorithm for m=4!

but there exists better than LS

(for ex. Woeginger, Galambos, 1993)

semi online scheduling

The first paper:

- Kellerer, Kotov, Speranza and Tuza (97), three models (m=2, makespan is minimized, plus):
- We know the total size of jobs (also by G. Zhang)
- or we can use a buffer of size K
- or we can make two schedule, and finally chose the better one
- For all of them: opt. Comp ratio 3/2→4/3
- Further models: Decreasing sizes, we know the value of OPT, etc., for example:

REASSIGNMENT (a bit later)

Reassignment in „real life”

- Seasons (winter, spring,…)
- Working places, living places (Toronto is very nice!)
- in politics: in every 4 years
- in economics: „big fish eats small fish”
- in personal habits: young man likes icecream,

  older man likes beer, stew, (more beer)…

in online models:

- our decisions must be made just in time.

(just when something happens we must make our decision)

- we cannot change our decisions later

reassignment: (some kind of semi online):

- We have some time to think, (we are allowed to delay decisions = reordering buffer), or
- we can change (a bit) later our decisions

rearrangement in bin-packing:

G. Galambos and G. J. Woeginger, Repacking helps in bounded space on-line bin-packing, Computing, Volume 49, Number 4,

- on-line bounded-space bin-packing problem where:
- repacking the items within the active bins is allowed.
- the 1.69103 lower bound of Lee and Lee for the worst case ratios of bounded-space approximation algorithms still applies.
- A polynomial time approximation algorithm is presented, that reaches the best possible worst case ratio matching the Lee and Lee lower bound while using only *three* active bins.
- (and other papers)

rearrangement in scheduling:

- the first model considering reassignment is among the first semi online models:

having a (reordering) buffer of size K

- Kellerer, Kotov, Speranza, Tuza: Semi online algorithms for the partition problem, *Op.Res Letters* 21 (1997), 235-242.

Results: -three semi online versions of P2 II C_max,
- three optimal algorithms with comp. ratio C=4/3,
(the pure online comp. ratio is C=3/2 (LS))

the buffer model is also treated in:

- G.C. Zhang, A simple semi-online algorithm for P2||C_max with a buffer. Information Processing Letters, 61 (1997), 145-148.

there are also some (new?) models:

- bounded migration by

N. Sivadasan, P. Sanders, M. Skutella, Online scheduling with bounded migration, Math. Oper.Res. 34 (2) (2009) 481-498.

(for any job there are two parameters, the size and a rearrangement parameter, and also given a global rearrangement factor β,…a **different type** of rearr.

so we do not deal with this model now,

on the other hand it is in fact interesting)

- REAR(K), (rearr. at any time) defined by

G. Dosa, Y. Wang, X. Han, H. Guo, Online scheduling with rearrangement on two related machines, Theoretical Computer Science, 412(8-10): 642-653 (2011)

- and three further models of

Z. Tan, S. Yu, Online scheduling with reassignment, Oper.Res.Lett. 36(2) 2008, 250-254., as :

- K jobs can be rearranged at the end of the sequence (*after the sequence ended*)
- The last job of *any machine* can be moved to the other machine (*after the sequence ended*)
- The last K jobs of the *sequence* can be moved (*after the sequence ended*)

now we consider only three models, only in case of 2 uniform machines (i.e., the second machine is faster)

- BUFF(K), the buffer problem

  G. Dosa, L. Epstein, Online scheduling with a buffer on related machines, J.Comb.Optim. 20(2) 2010, 161-179.

- REAR(K), at any time (many times)

  G. Dosa, Y. Wang, X. Han, H. Guo, Online scheduling with rearrangement on two related machines, Theoretical Computer Science, 412(8-10): 642-653 (2011)

- REND(K), rearrangement only at the end

  A. Benko, X. Chen, G. Dosa, X. Han, Online scheduling with bounded rearrangement at the end, TCS, 2011.

  (the model is defined originally by Tan, 2008)

without reassignment and buffer (pure online case, i.e. K=0)

In this case, the tight competitive ratio is

$(2s+1)/(s+1)$ if $1 < s < 1.618 = (\mathit{sqrt}(5)+1)/2$

and

$(s+1)/s$ if $1.618 < s$

(L. Epstein, J. Noga, S. S. Seiden, J. Sgall, and G. J.Woeginger, Randomized Online Scheduling on Two Uniform Machines, *Journal of Scheduling*, 4(2):71–92, 2001.)

comparison between the models

- (we know that at least formally)

  REAR(K) is more flexible than BUFF(K)

  (=allows at least as good comp. ratio)

BUT

- Is REND(K) more flexible than BUFF(K) ?
- Is REND(K) more flexible than REAR(K) ?

since !!!!! *in REND(K)*

- we are allowed to make rearrangement only once
- but we know before the rearrangement that the sequence is over (We will see that any of the three models allows better comp. ratio than the pure only model)

Results 1. if $s \geq 2$:

- everything is quite simple
- K=1 is enough

(K=1 is better than K=0, but K>1 do not give much than K=1)

- The tight ratio is $C=(s+2)/(s+1)$

uniformly, for all three models!

(for K=0 the tight ratio is $(s+1)/s$)

Results 2. if $1 \le s \le 2$, and K=2

- this case is not very hard (but not simple)
- K=2 is enough

(K=2 is better than K=0, but K>2 does not give better than K=2)

- The tight ratio is

$C=(s+1)^2/(s^2+s+1)$ if $s \le 1.61 =$ *(sqrt(5)+1)/2*

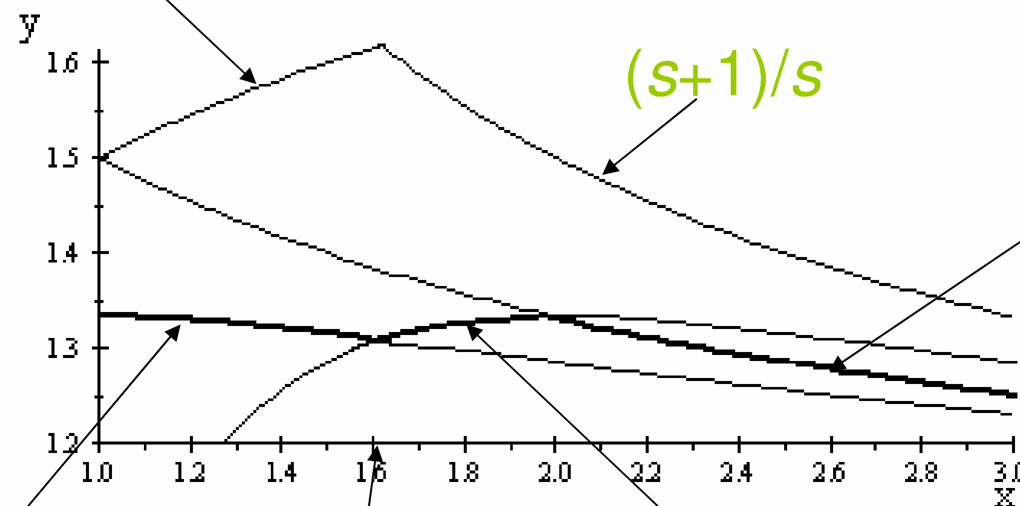
$C = s^2 / (s^2 - s+1)$ if $s \ge 1.61$

uniformly, for all three models!

so, the tight ratios:

(2*s*+1)/(*s*+1)

pure online (K=0)



(*s*+1)/*s*

(s+2)/(s+1),

tight for K≥1, from s=2

(sqrt(5)+1)/2

$s^2/(s^2 - s+1)$

$(s+1)^2/(s^2+s+1)$

for all three models!

tight for K=2 (or bigger)

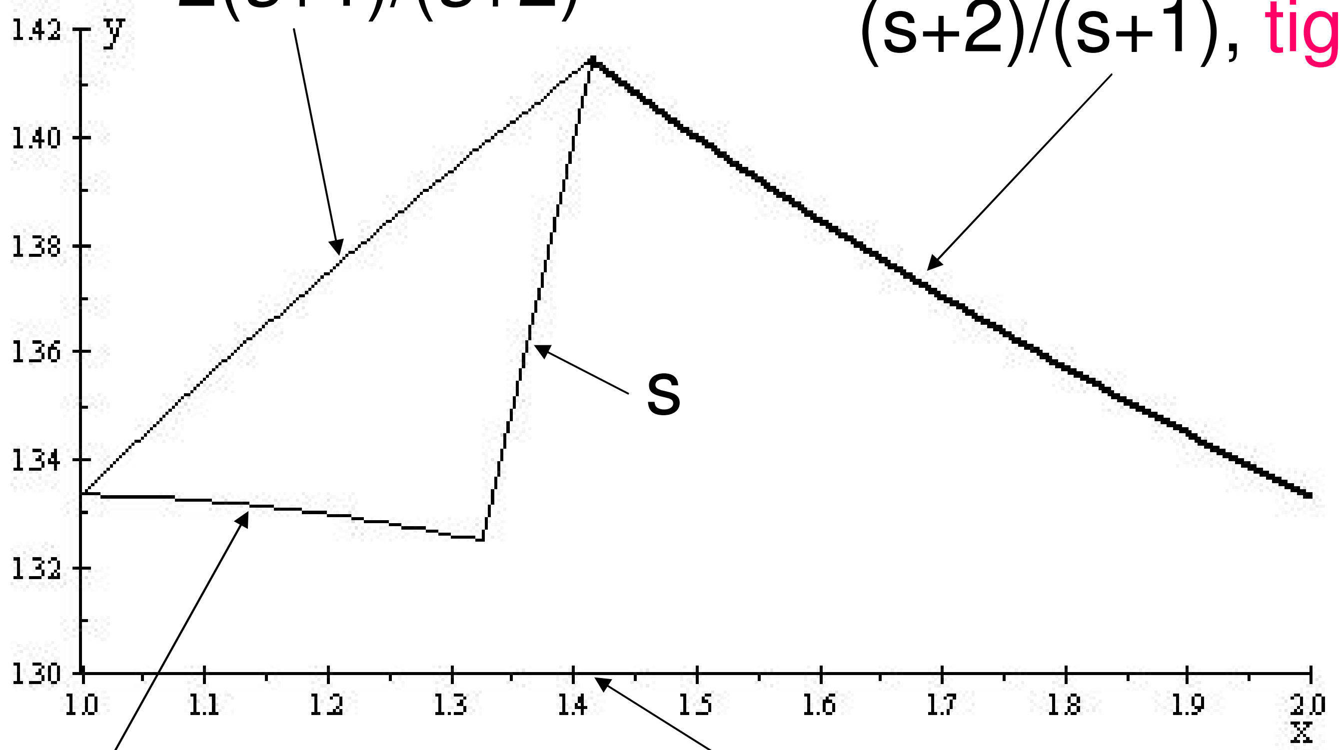
Results 3. the hard case: $1 \leq s \leq 2$, and $K=1$

- we actually do have results (in all three models), but it seems hard to get the tight ratios for the whole interval (in any of the three models)
- here the models differ from each other (regarding competitive ratio)
- we still have the best results so far (for all three models)

what we could prove, BUFF(1):

2(*s*+1)/(*s*+2)

(s+2)/(s+1), tight



s

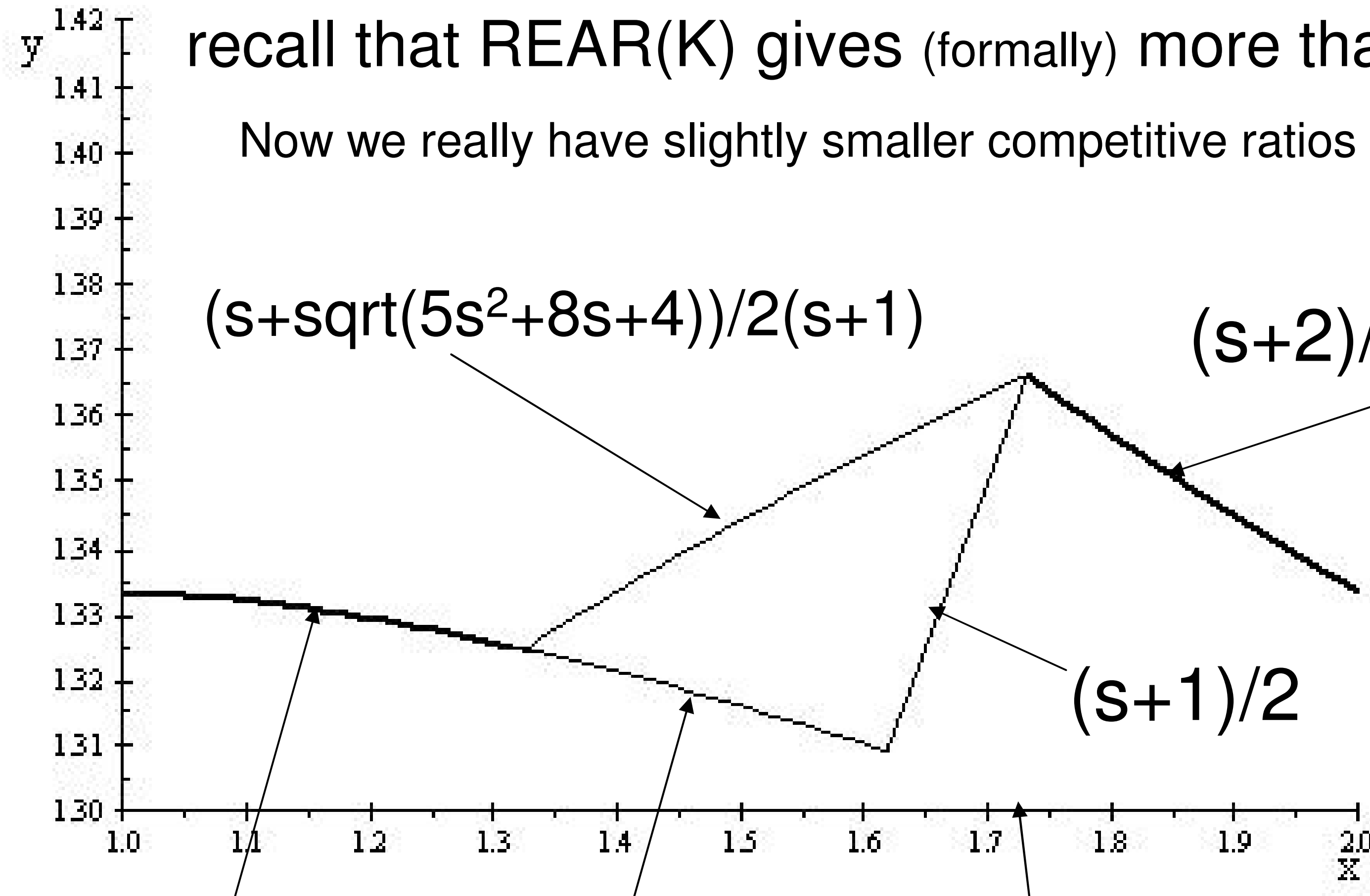
$(s+1)^2/(s^2+s+1)$ is still valid,
since it holds for any K

sqrt(2)

what we could prove, REAR(1):

recall that REAR(K) gives (formally) more than BUFF(K)!!

Now we really have slightly smaller competitive ratios somewhere



The third model: Two simple algorithms for REND(1), for both:

- We use the (classical online) lower bound LB: max{total size/(s+1), biggest job/s}
- We also use the allowed competitive ratio C(s)
- A schedule is feasible, if the desired competitive ratio is not violated, where the (increased) makespan is compared to the lower bound LB

(the makespan is not bigger than C-times LB)

ALG1 for REND(1)

- Assign the incoming job to M1 if so the (temporary) schedule is feasible, otherwise to M2.
- REASSIGNMENT: If the final schedule is not feasible, move a job from M2 to M1 so that the makespan decreases as much as possible
- It is $(s+2)/(s+1)$-comp. for any $s \geq 1$
- so for $s \geq 2$ it is optimal.

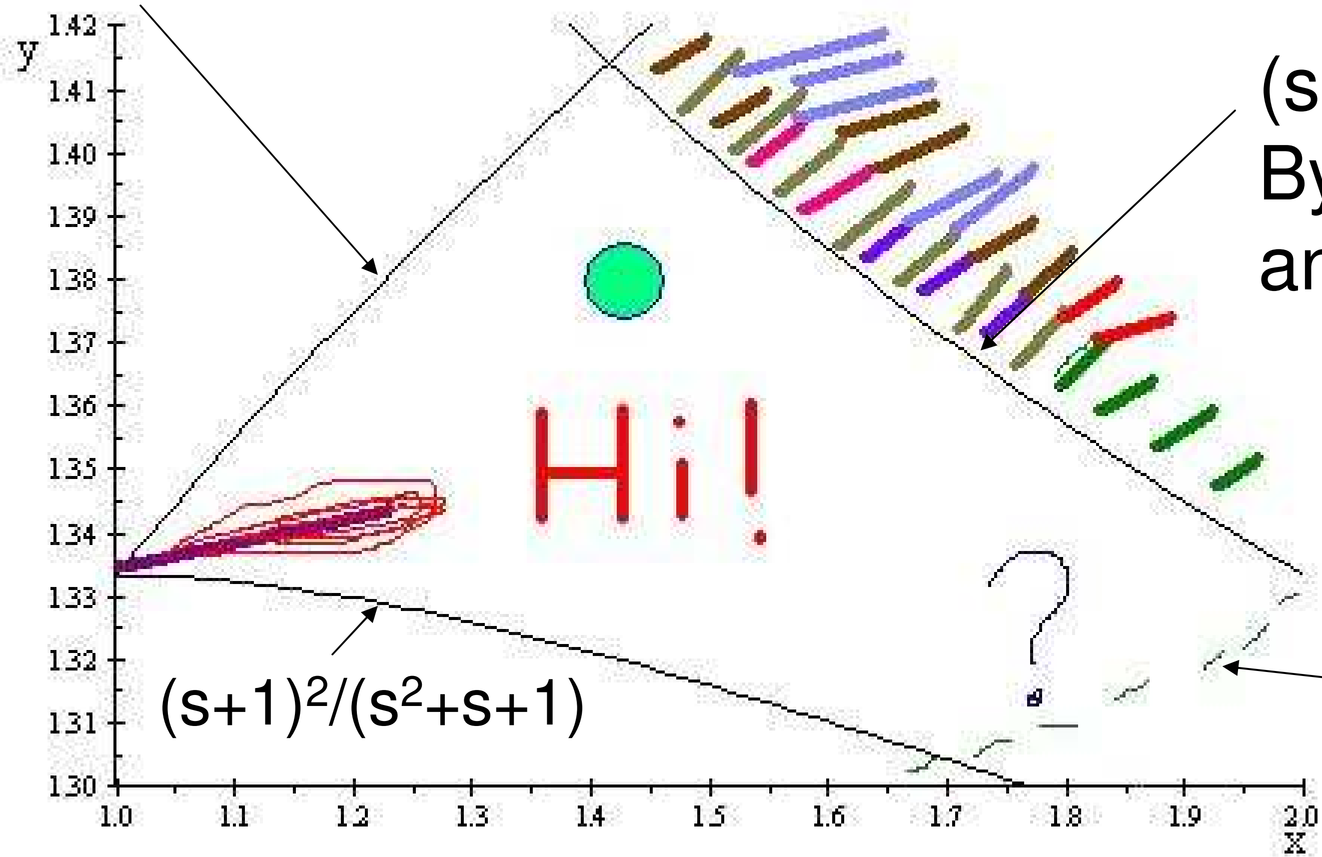
ALG2 for REND(1)

- Assign the incoming job to M1 if so the (temporary) schedule is not feasible, but moving the biggest job from M1 to M2 the schedule becomes feasible, otherwise to M2.
- REASSIGNMENT: If the final schedule is not feasible, move the biggest job from M1 to M2 if so the makespan decreases
- It is (s+2)/(s+1)-competitive only if 1≤s≤2, but
- We know in advance what job will be moved
- And it is also 2(s+1)/(s+2)-competitive

what we could prove, REND(1):

(only preliminary results so far, but these are the best results)

$2(s+1)/(s+2)$ by ALG2



$(s+2)/(s+1)$ By ALG1 and ALG2

what is the case here?

good lower bounds? really challenging task,

the same upper bounds as in BUFF(1)!

previous constructions do not work!

but now we nowhere know whether they are tight or not

Algorithms (some statistics)

- buffer model BUFF(K):

  4 algorithms on 9 pages

- rearrangement at any time REAR(K)

  1 algorithm, the analysis is on 5 pages through 5 Lemmas, 8 Observations and 3 Claims

- rearrangement at the end REND(K)

  3 algorithms, the proofs are on ≈20 pages through 11 Lemmas, 11 Observations, many Cases

~~YANK THOU!~~ THANK YOU!