College admission in practice

Claire Mathieu

joint work with Hugo Gimbert

Schools
looking for students

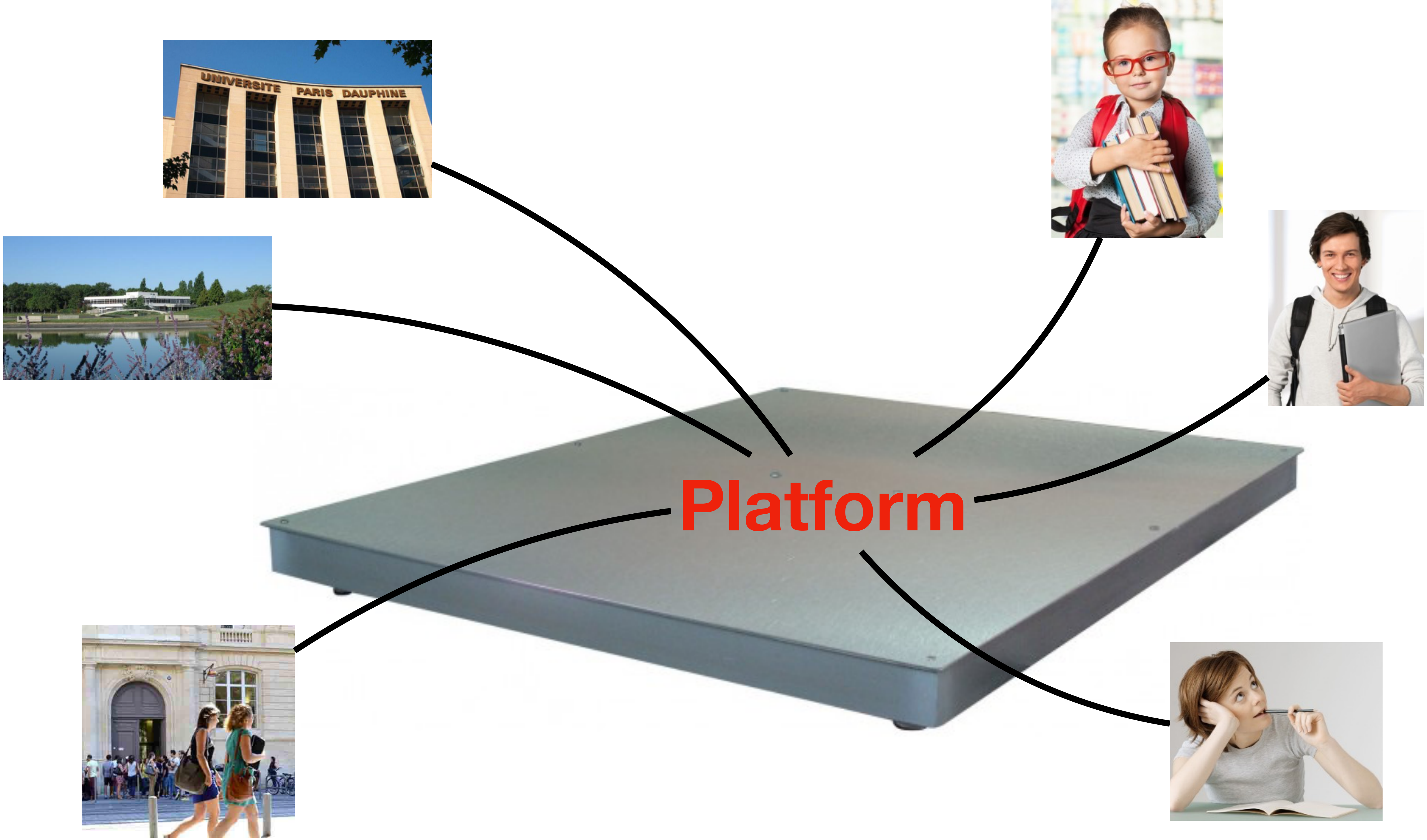
Students
looking for schools

How to match students to schools?

College admissions in 2018 in France

- over 800000 applicants
- over 10000 degrees

Automatic processing is a necessity


UNIVERSITE PARIS DAUPHINE
Platform

Platform design

Let the market rule?

Each school advertises its openings
Each student looks around
Offers happen
Everyone has their own deadlines

"Had I known…" **"Had I known…"**

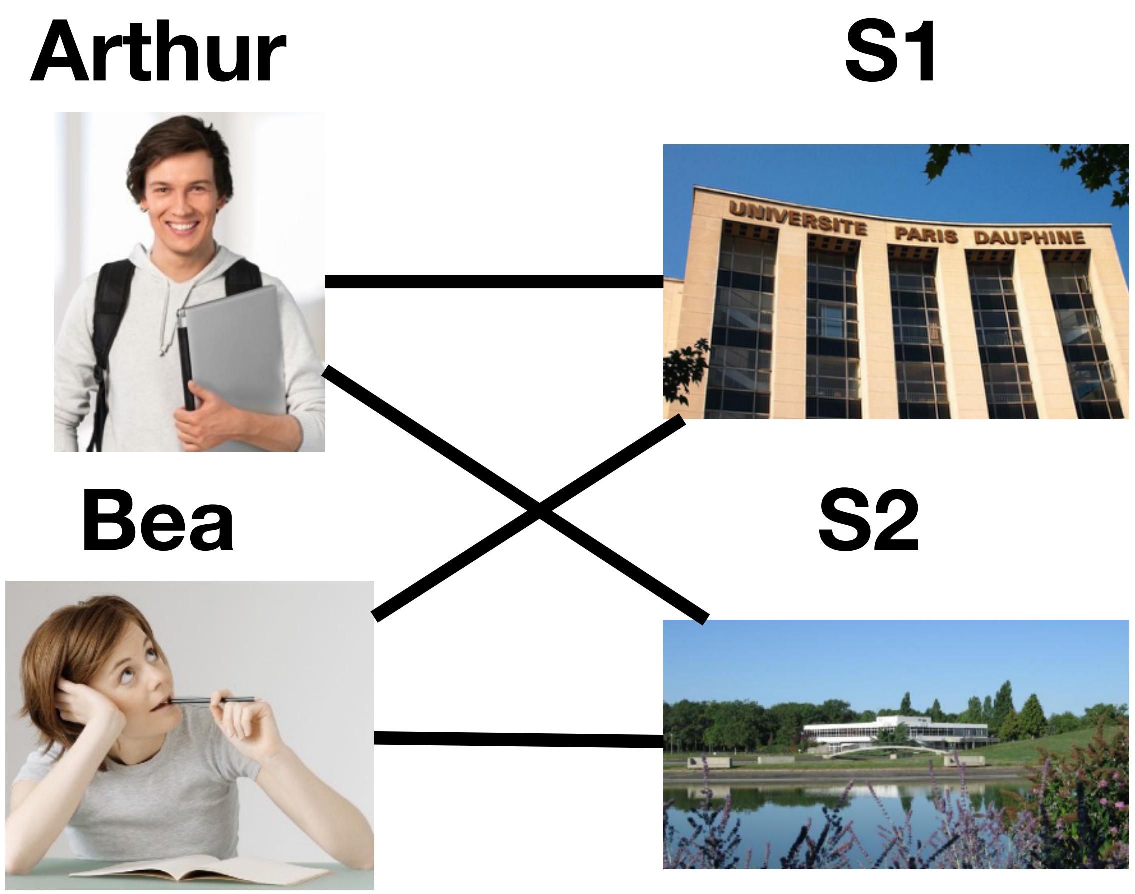
regrets, inefficiency, chaos, instability

Some order is needed

Tool #1: Common deadlines

1. Students apply before a common deadline
2. Schools look at applications and make offers before a common deadline
3. Each student accepts his/her best offer

Still inefficient



1. Arthur and Bea apply to S1 and S2
2. S1 and S2 both make offers to Bea
3. Bea chooses S1

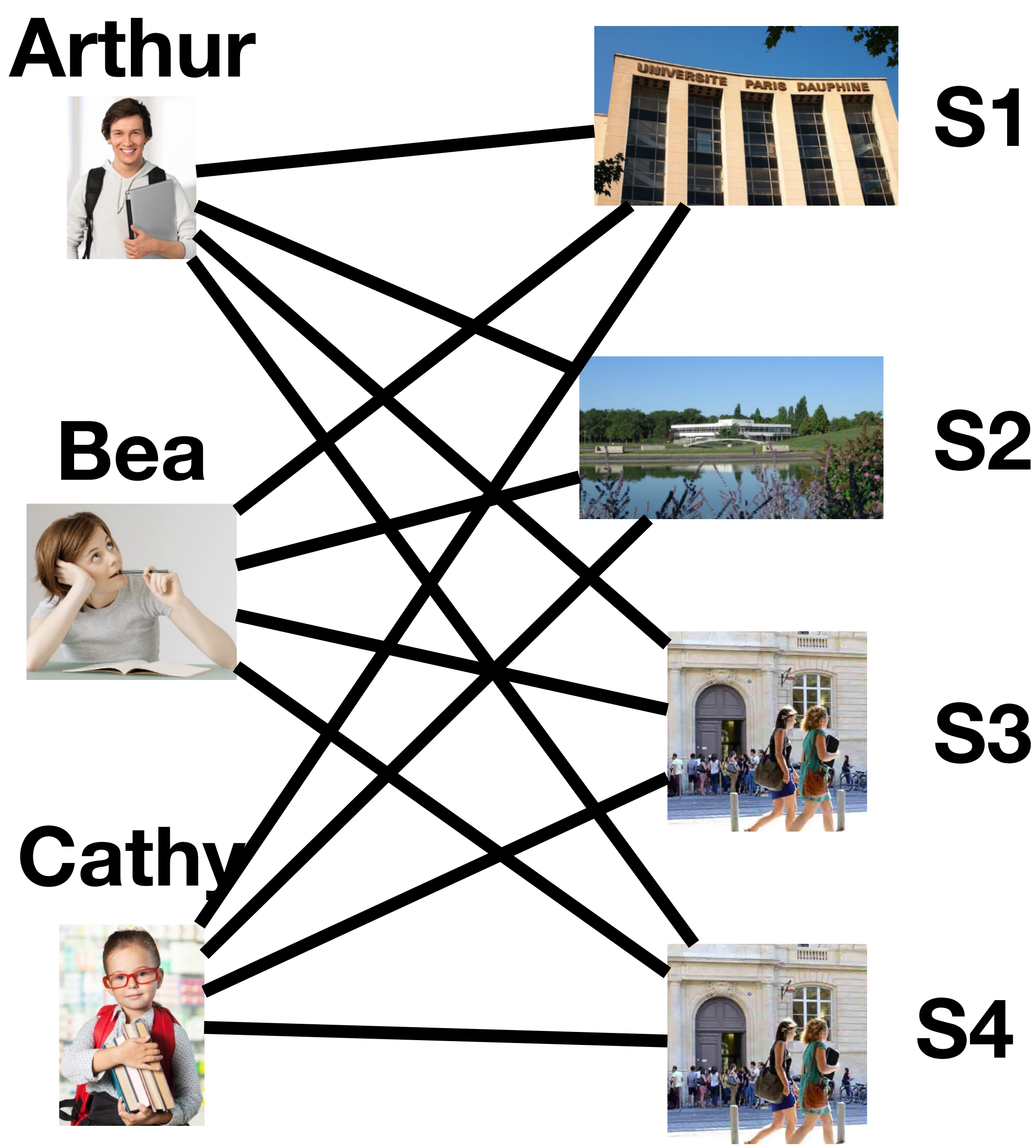
Arthur has no offer,
S2 recruits no student:
Regrets!

Tool #2: Rounds

1. Students all apply before a common deadline
2. Schools look at applicant folders and all make their offers before a common deadline
3. Each student accepts his/her best offer
4. Schools with remaining slots make offers to remaining students before a deadline
5. Each student accepts his/her best offer

And repeat 4+5 as needed…

Still inefficient



1. Arthur and Bea apply to S1,S2,S3,S4
2. S1,S2 make offers to Bea, S3,S4 to Arthur
3. Bea chooses S1, Arthur S3
4. S2,S4 make offers to Cathy who chooses S4

Bea would have preferred S4,
S4 would have preferred Bea:
Regrets!

Bea: "Had I known, I would have said no to S1,S2 and waited to get an offer from S4 on the second round"
S4: "Had I known, I would have skipped Arthur and started by making an offer to Bea while she was still available"

Tool #3: Allow change of mind

1. Arthur and Bea apply to S1,S2,S3,S4
2. S1,S2 make offers to Bea, S3,S4 to Arthur
3. Bea chooses S1, Arthur S3
4. S2,S4 make offers to Bea even though she is already assigned. Bea changes her mind and chooses S4
5. S2 makes offer to Cathy who accepts

No regrets!

The Gale-Shapley algorithm

Input :
Each school ranks students
Each student ranks schools
Each school has a capacity

Iterate:

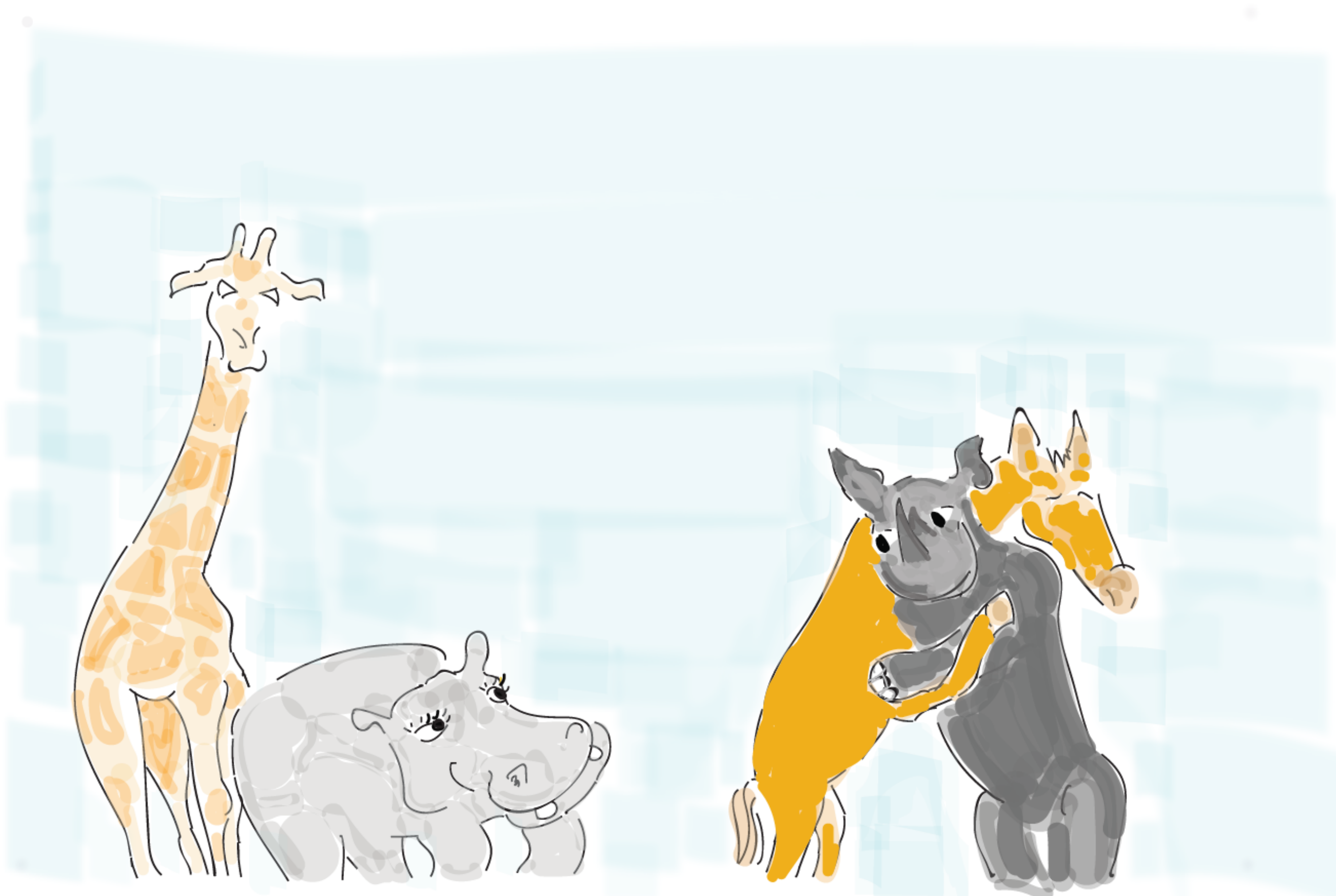
1. Each school sends an offer to next students on list, up to (residual) capacity
2. Each student looks at new offers plus previously accepted offer (if it exists), and rejects all except their favorite, which they tentatively accept.

Condition: when nothing happens for one iteration
All tentative accepts become final

Properties

Polynomial time

Output has no *blocking pair*:
(student,school) who
would have preferred each other
to what they have



The *other* Gale-Shapley algorithm

Input :
Same

Iterate:

1. **Each student sends an application to next school on their list**
2. **Each school looks at new candidates plus previously accepted candidates, and rejects all except their favorites, which they tentatively accept up to capacity.**

Condition: **same**

Properties : same +
no student has an incentive to lie

Gale-Shapley in practice?

Comparing the two versions

Almost identical in practice: almost every student (> 99.9%) has the same school in both (2017 data)

Uncertainties in practice

Students' ranking is uncertain…
School capacity is uncertain…
Set of students is uncertain…
Offers might be conditional…

Handling uncertainties with time

- Do not ask for ranking until offer in hand
- Update assignment daily to incorporate changes in capacities or set of students

Input :
Each school ranks students
Each school has a capacity

Iterate daily starting on May 22:

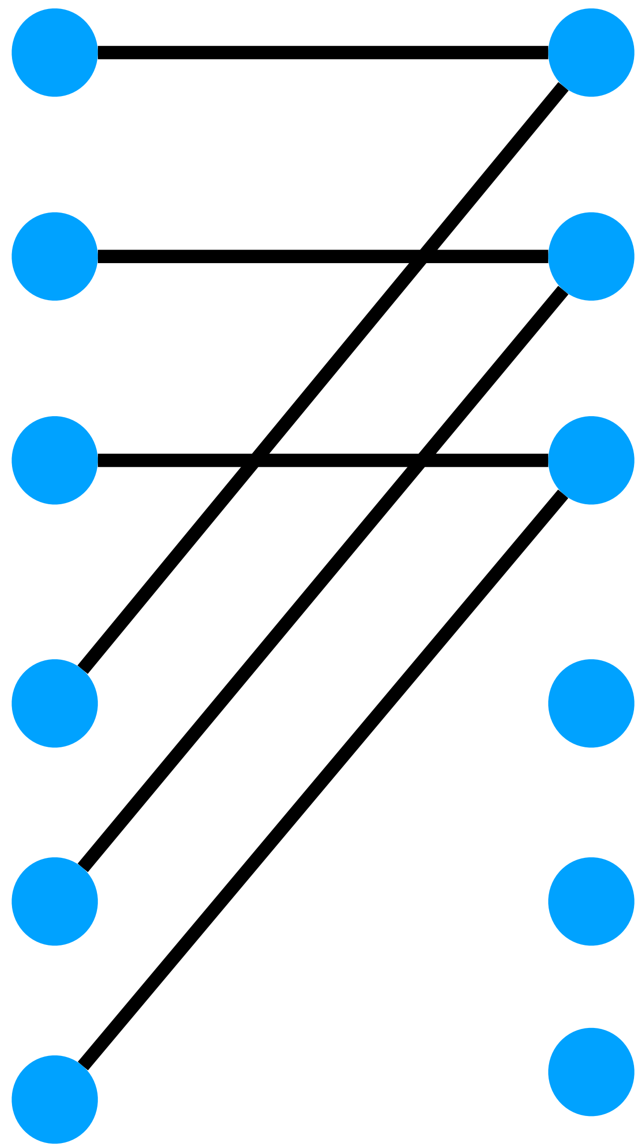
1. **Each school sends an offer to next students on list, up to current capacity**
2. **Each student looks at new offers plus previously accepted offer (if it exists), and (within 3 days) rejects all except their favorite, which they (tentatively) accept.**

Condition: when school starts (on Sept 5)
All tentative accepts become final

How long until convergence of main procedure?

If every student makes 1 wish:
1 iteration

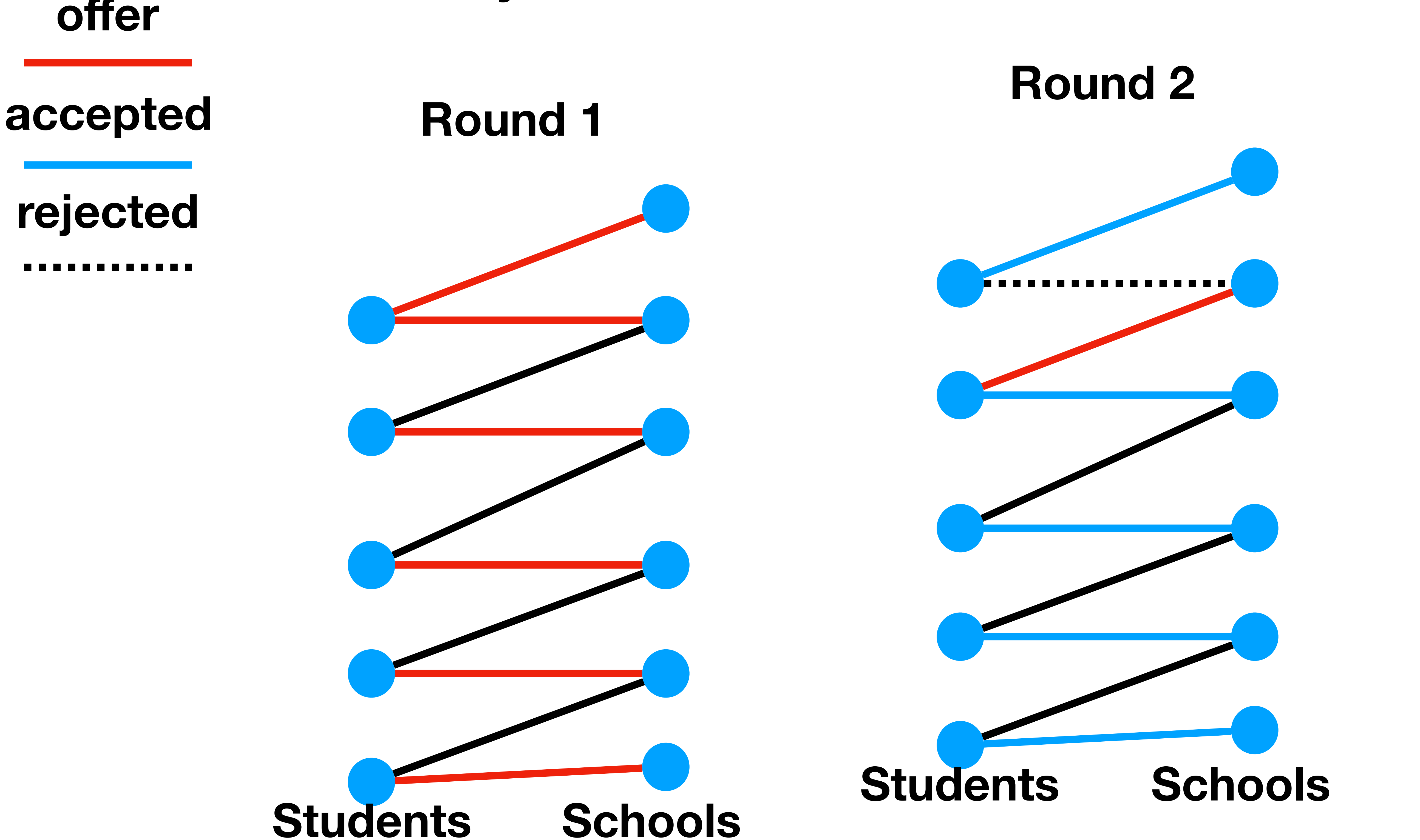
Students



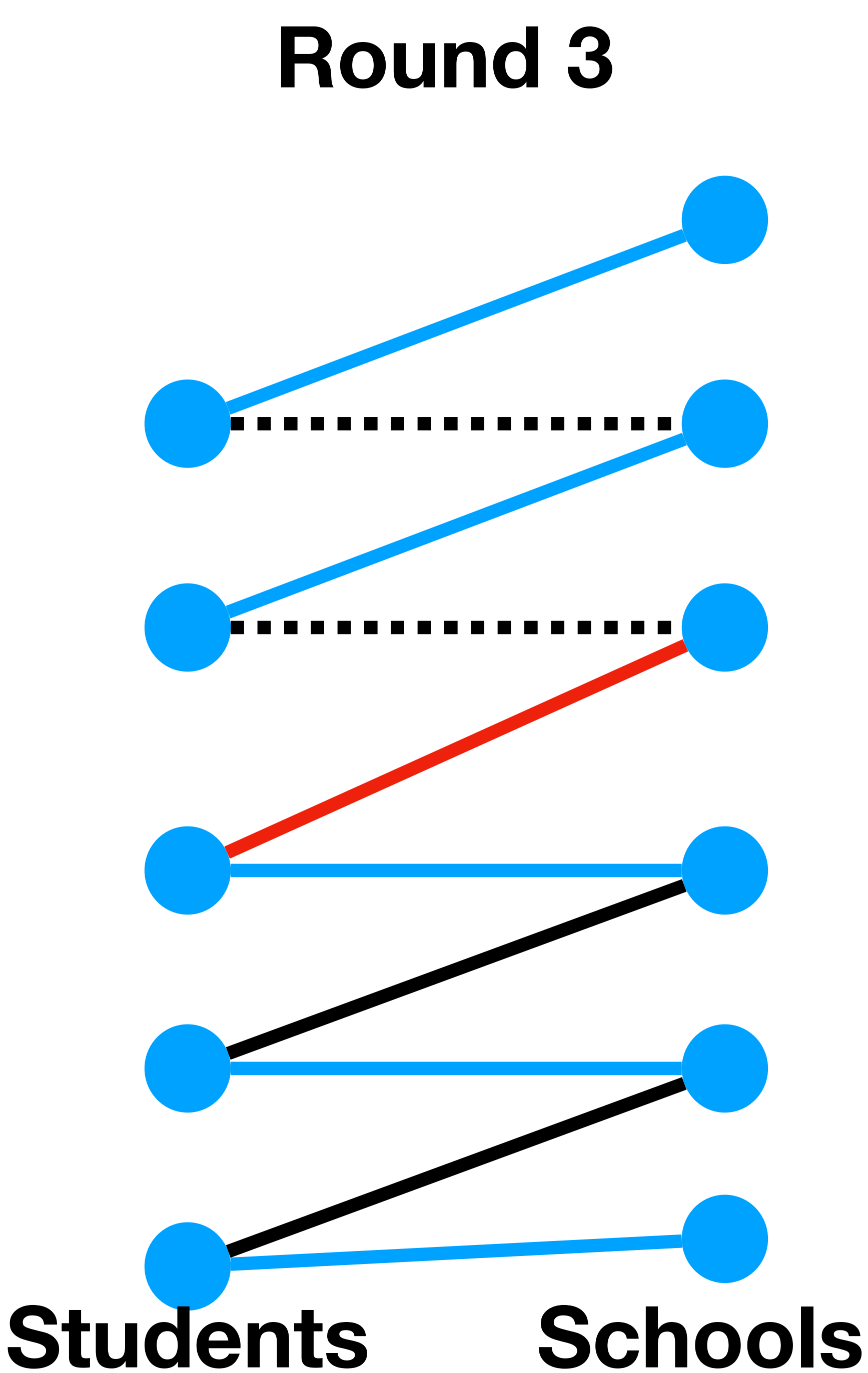
Schools (capacity 1)

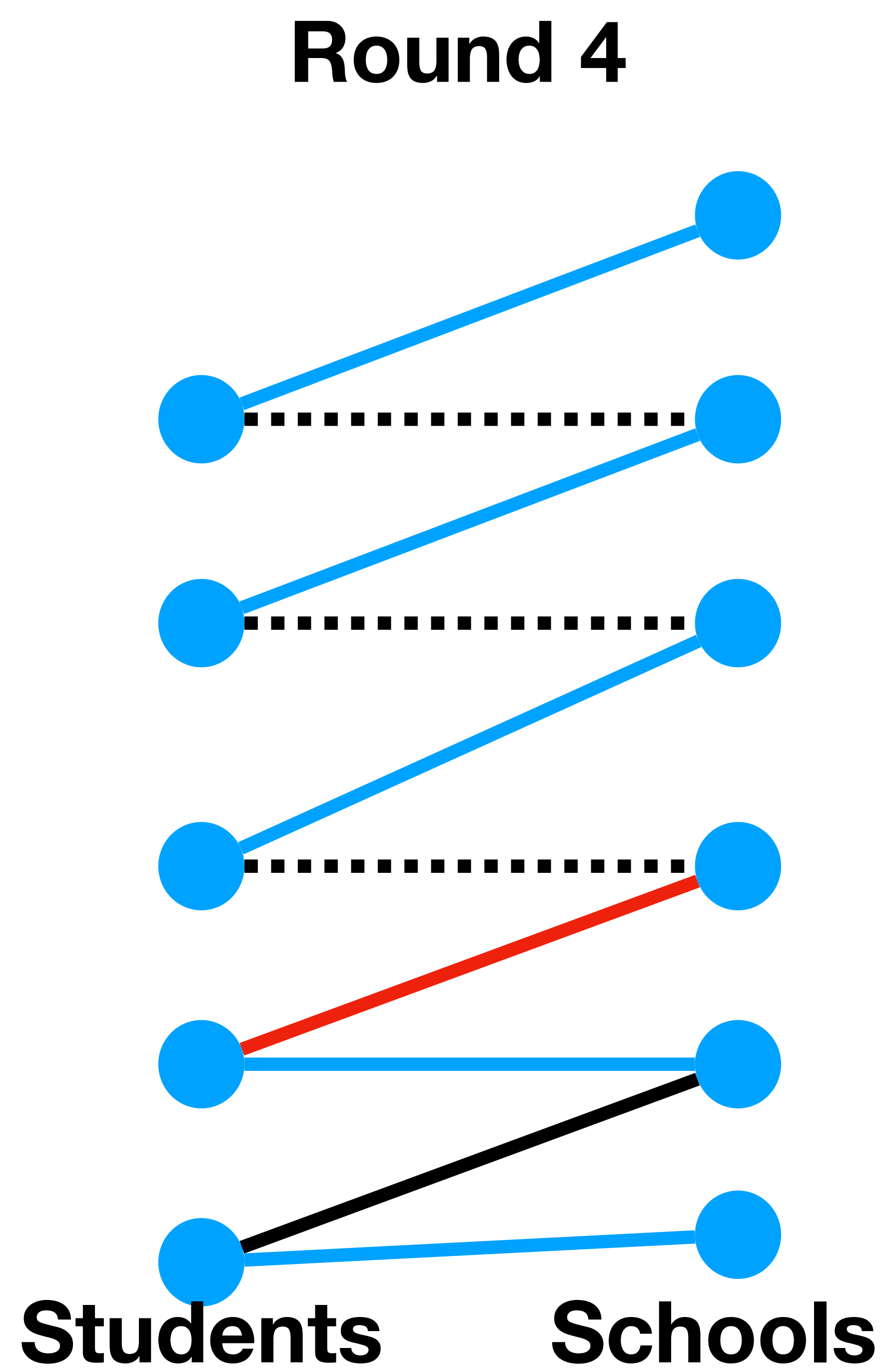
Schools (capacity 1)

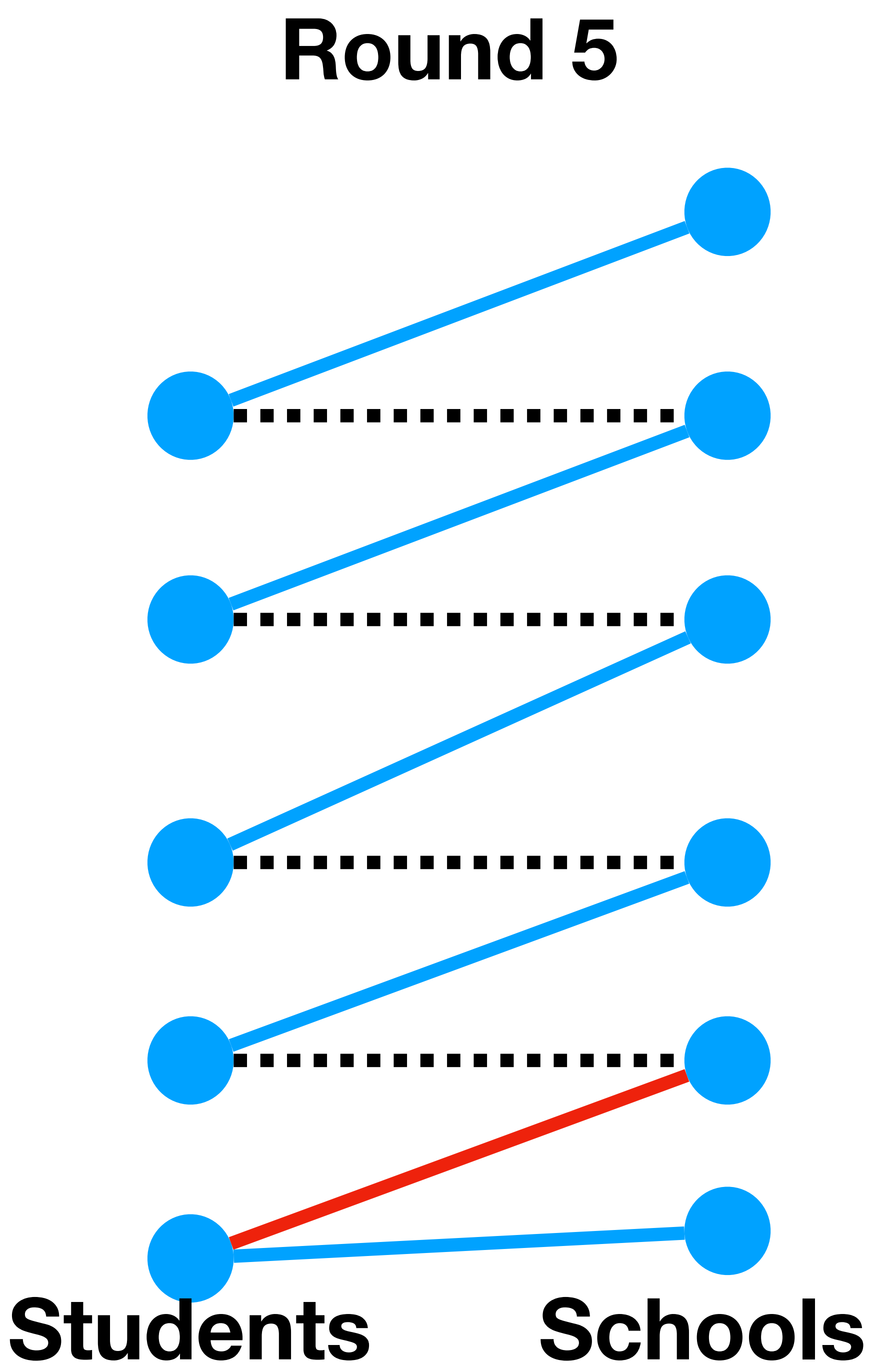
If every student makes 2 wishes

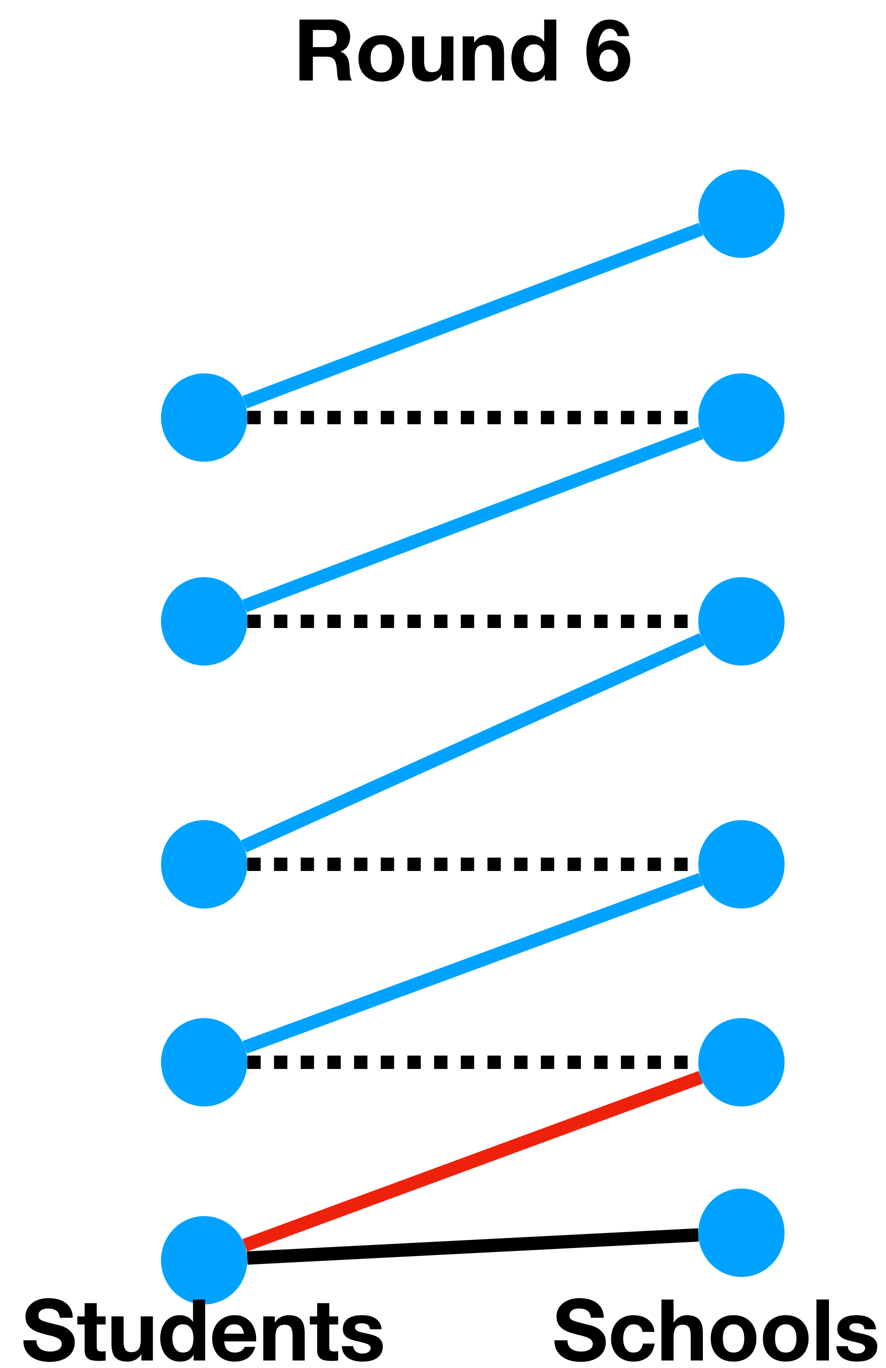


Schools (capacity 1)









Number of iterations can be #edges…

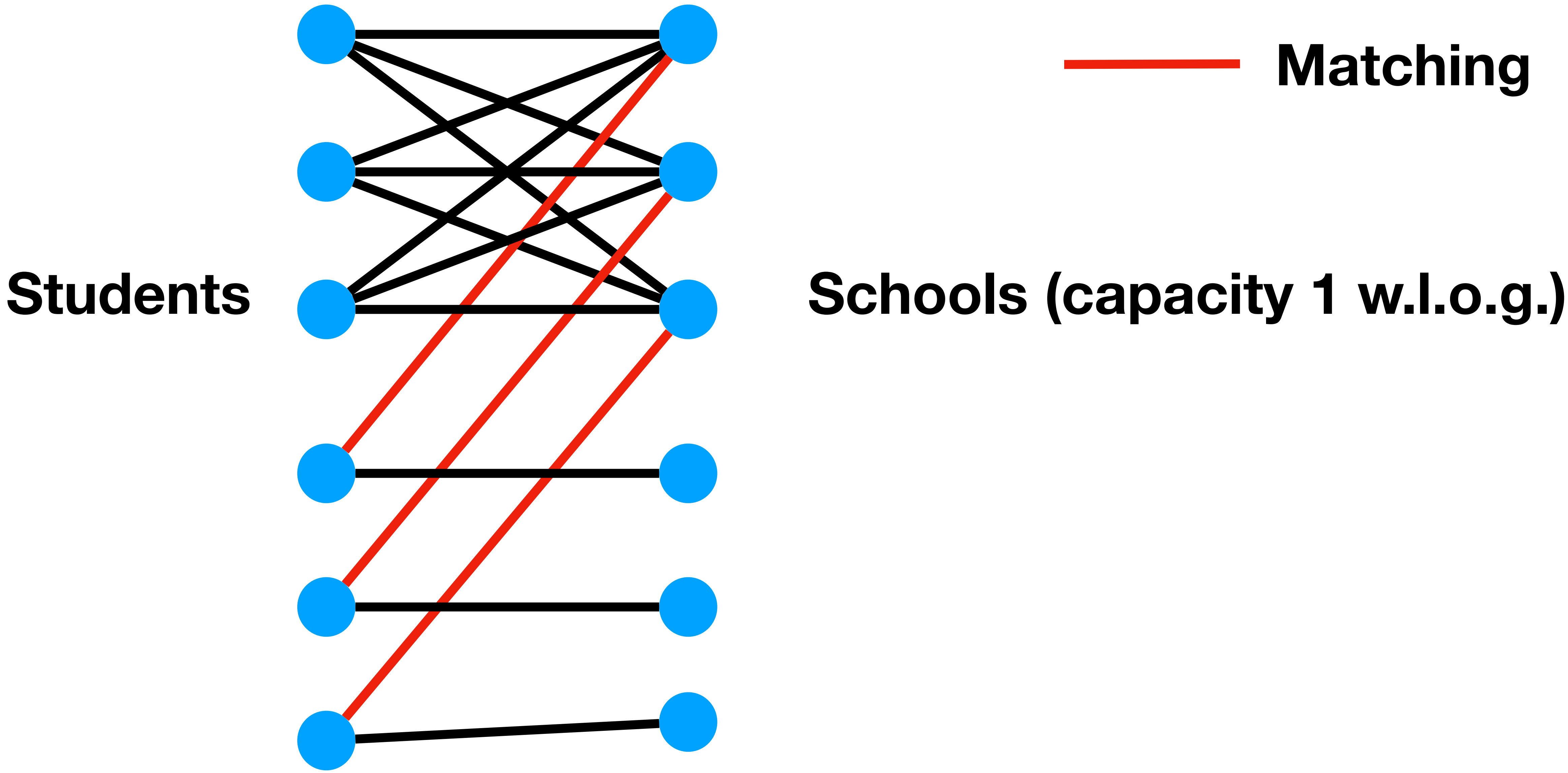
Gale-Shapley: how long until convergence?

Worst case:
convergence
is
quadratic

Simulations:
convergence
by mid-summer,
mostly

Observations:
almost
no action
by end of July

How many candidates are eventually assigned?



Number of students assigned is

- **at most maximum matching**
- **at least maximal matching**

What to do with leftover candidates

An ad hoc complementary procedure assigns leftover students to leftover slots

2018 final result

583 000 registered in higher education through Parcoursup main and complementary procedures : 27000 more than in 2017

Three algorithmic questions

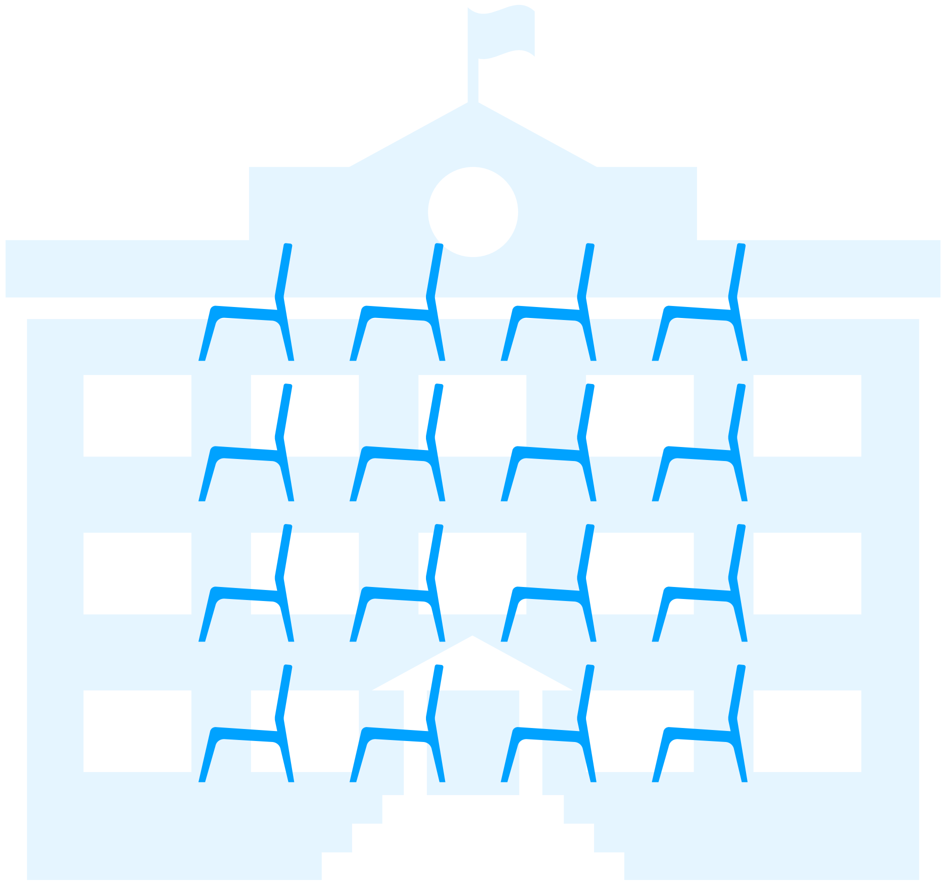
On top of the main procedure

1. Coupling school assignment with assignment of dorm beds
2. Quotas of low-income students
3. Quotas of low-income and of local students

Dorm beds

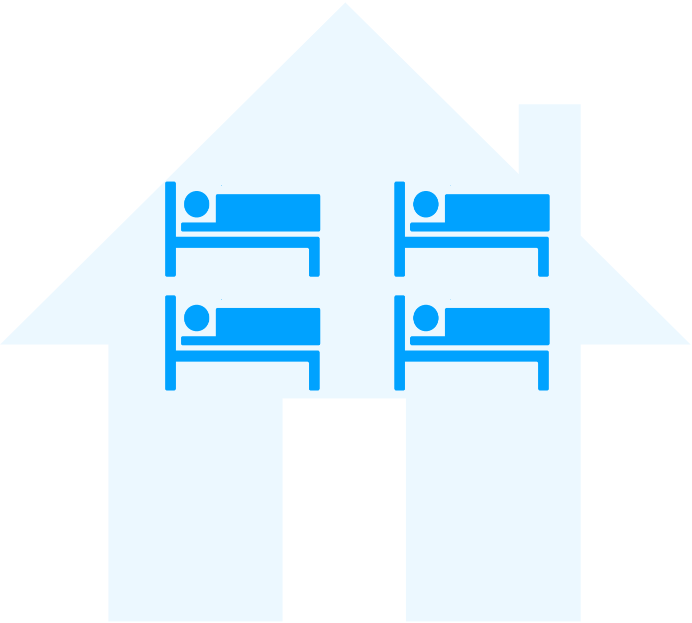
Two rankings

School ranking : A B C D E F



Academic criteria

Dorm ranking : C F A E B D



Social and geographic criteria

What if a candidate says:
“I will only come
if I get a dorm bed”

Risks

Strategies:
an applicant requires a dorm to increase his chances of getting it

Answer:
each applicant can make two applications

- **school with dorm**
- **school without dorm**

They are treated independently of each other, s.t. capacity constraints

A student may receive an offer “school without dorm” and at some later point “school with dorm”

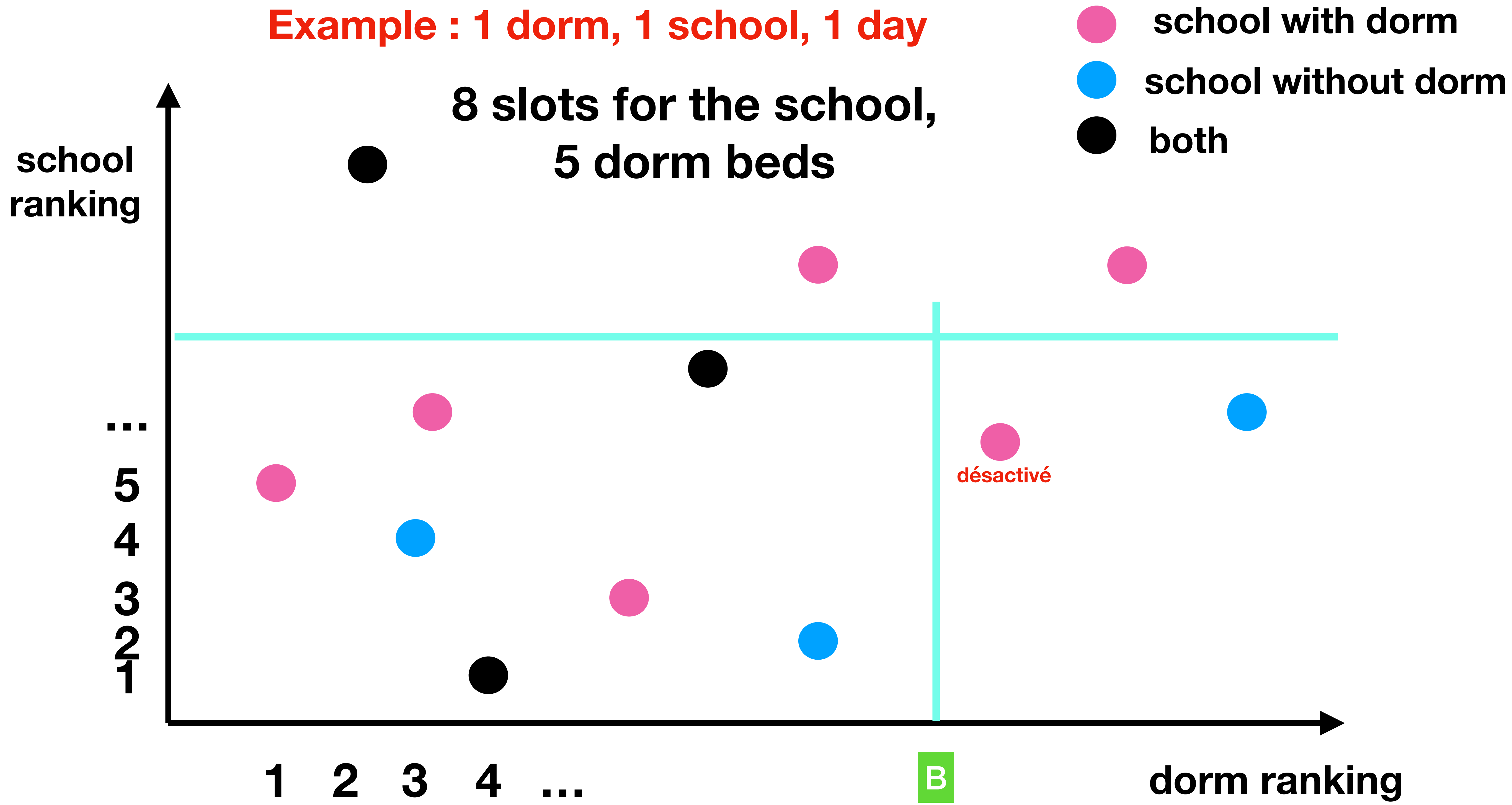
Desired properties

Must not exceed school capacity
Must no exceed dorm capacity
Fair:

- **If Alice asks for "school without dorm" and Alice precedes Barbara in school ranking, then Alice should get an offer before Barbara**
- **If Barbara asks for "school with dorm" and Alice precedes Barbara in both rankings, then Alice should get an offer before Barbara**

Aim to fill school and dorm to capacity

Example : 1 dorm, 1 school, 1 day



- Temporarily deactivate applicants requiring dorm, whose dorm rank is >B
- Offer the school to the first 8 applicants in school order
- Offer the dorm to those among them whose dorm rank is at most B

Choose B (max) so that the output offers 5 dorm beds.

General case

- Many schools, many dorms, many days
- Several dorms for the same school (men, women,…)
- Several schools share the same dorm

Each day:

Given dorm thresholds B1, B2, …

1. Temporarily deactivate application if dorm rank > dorm threshold
2. Offer each school i to the first (residual capacity) remaining applicants in school order
3. Offer each dorm j to all applicants s with an offer from school and whose dorm rank is at most Bj

Some dorm capacity may be exceeded.
To respect dorm capacities:

Starting from B1,B2,… very large

Repeat

1. Try above algorithm
2. If it fails, decrement some Bj s.t. j exceeds capacity

Until feasible

Theorem

Result does not depend on choice of threshold to decrement.
Final Bj= max possible for all j.

Quotas of low-income students

From law to specification

The law, in French

“l'autorité académique fixe un pourcentage minimal de bacheliers retenus bénéficiaires d'une bourse nationale de lycée, en fonction du rapport entre le nombre de ces bacheliers boursiers candidats à l'accès à cette formation et le nombre total de demandes d'inscription dans cette formation”

The law, in (ambiguous) Math

for school c, at least 25% of low-income students

The law, in (ambiguous) Math

for school c, at least 25% of low-income students:

↓

The law, in unambiguous Math

If the school makes k offers
then either:
at least k/4 offers to low-income students
or:
all low-income students got offers

Note: guarantees on opportunity

Algorithm

Modify school ranking greedily

school ranking

NNBN NNBN NNNN NNNN BNNN

Low-income quota algorithm: move low-income students up the list so that any prefix of the list satisfies the quota

BNNN BNNN BNNN NNNN NNNN

order of proposals

The legal constraint

(*) If the school makes k offers
then either:
at least k/4 offers to low-income students
or:
all low-income students got offers

Theorem:
Output
- **respects (*)**
- **is closest to the school ranking given (*)**

Proof

quota 25%

1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2
x x x L x x x x x | L x x x x x x x x x x x L L L L L x x x x x x

25% of 9 = 3: need 3 L's in first 9 letters
already have 1, so need 2 more
so: at least 2 L's must cross line

Greedy: exactly 2 L's cross line
Total displacement(Greedy) meets LB ✓

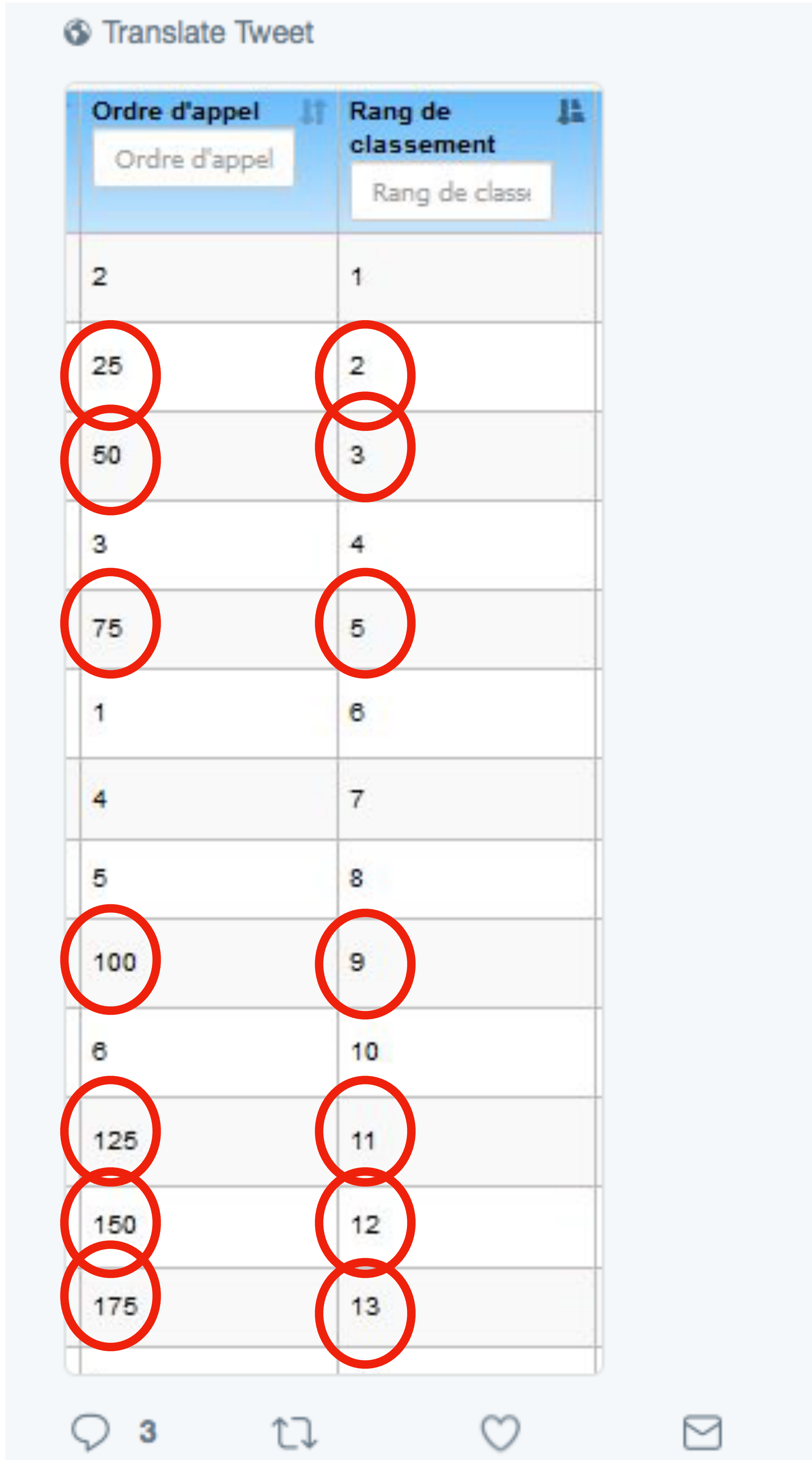
Quotas of local students

If the school makes k offers
then
either:
at least 98%k offers to local students
or:
all local students got offers

Algorithm: modify ranking greedily
for all k, at least 98% of first k students are local
until we're out of local students

Similar to low-income algorithm, yet, very different impact!

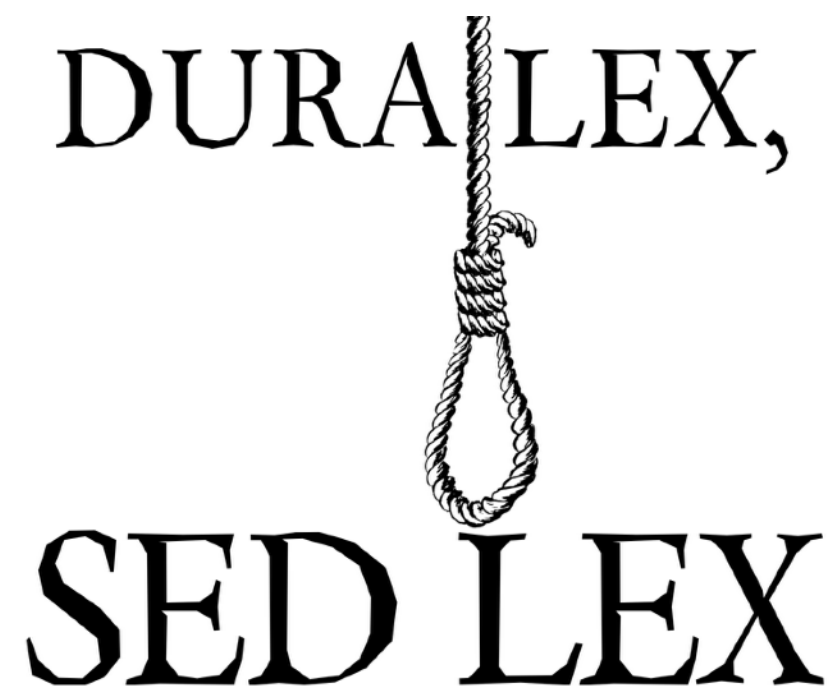
quota:
at most 4%
non-local
students


Translate Tweet

Ordre d'appel	Rang de classement
2	1
25	2
50	3
3	4
75	5
1	6
4	7
5	8
100	9
6	10
125	11
150	12
175	13

3

The school ranking
may be
completely modified



Both quotas

Higher authority:

quota:
at most 4% non-local students,
at least 25% low-income students

potential problem!...

Rule:
In case of conflict between quotas,
the low-income quota has priority

Algorithm for two quotas

For each k: if both quotas are currently critical then:

- **try to take next low-income local applicant,**
- **or else next low-income applicant,**
- **or else next local applicant**
- **or else next applicant**

Conclusion

What Theory brings to the table:

- Algorithmic techniques and representations
- Rigorous perspective
- Proofs!

The advantage of simplicity cannot be overrated