ALGOLAB TUTORIAL #9

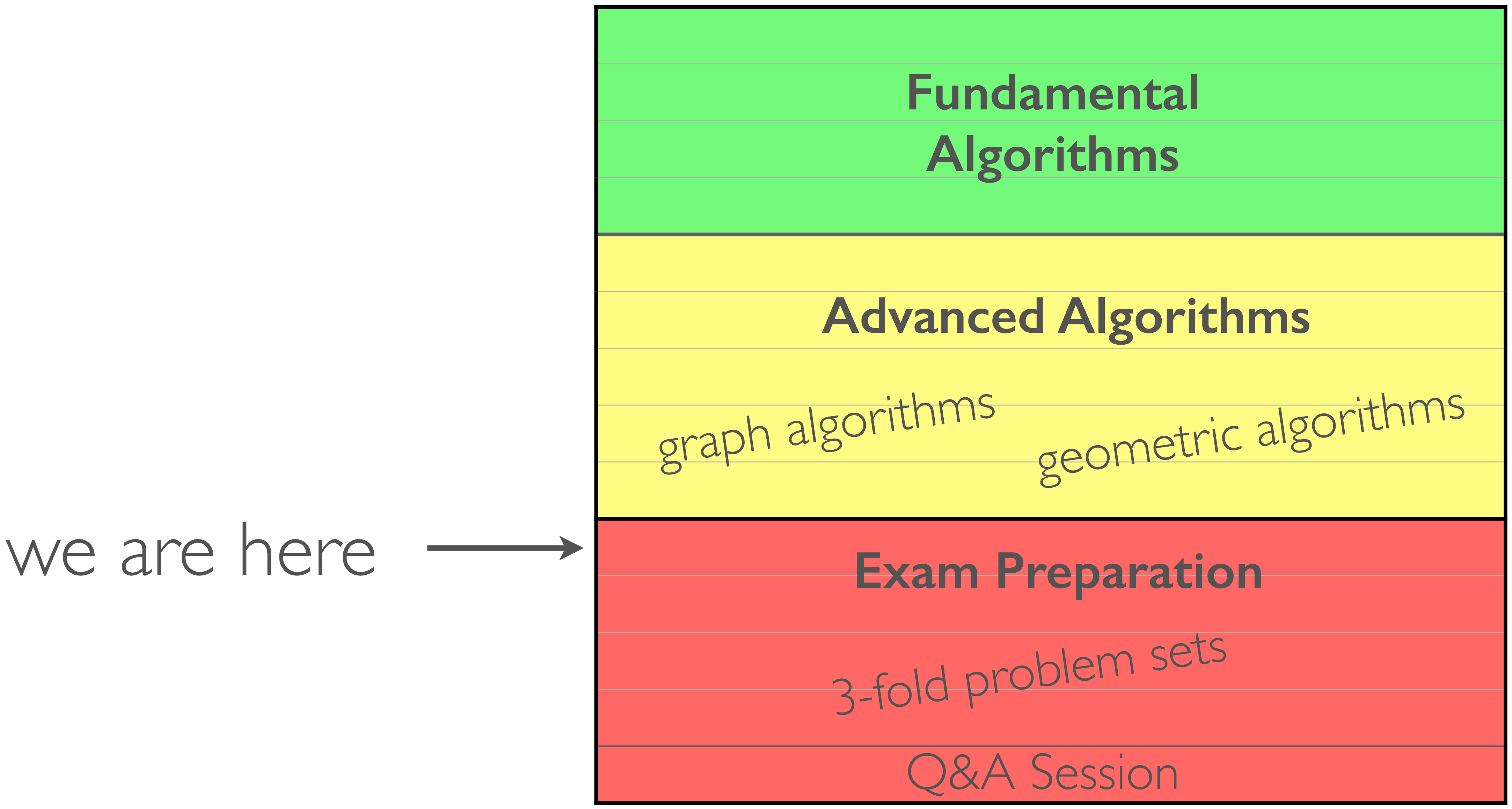
Exam Preparation Week I

Michael Hoffmann <hoffmann@inf.ethz.ch>

Contents

- Information about the test exam
- How to solve Algolab problems (meta-guidelines)
- Discussion of Problems (Firsthit, H1N1)

ALGOLAB TIMELINE



Exam (afawk): Jan 27 and Feb 1, 2016, 13-19.

TEST EXAM

Date/time: Tu, **Dec 8, 2015, 17:00-19:15**, ETH HG.

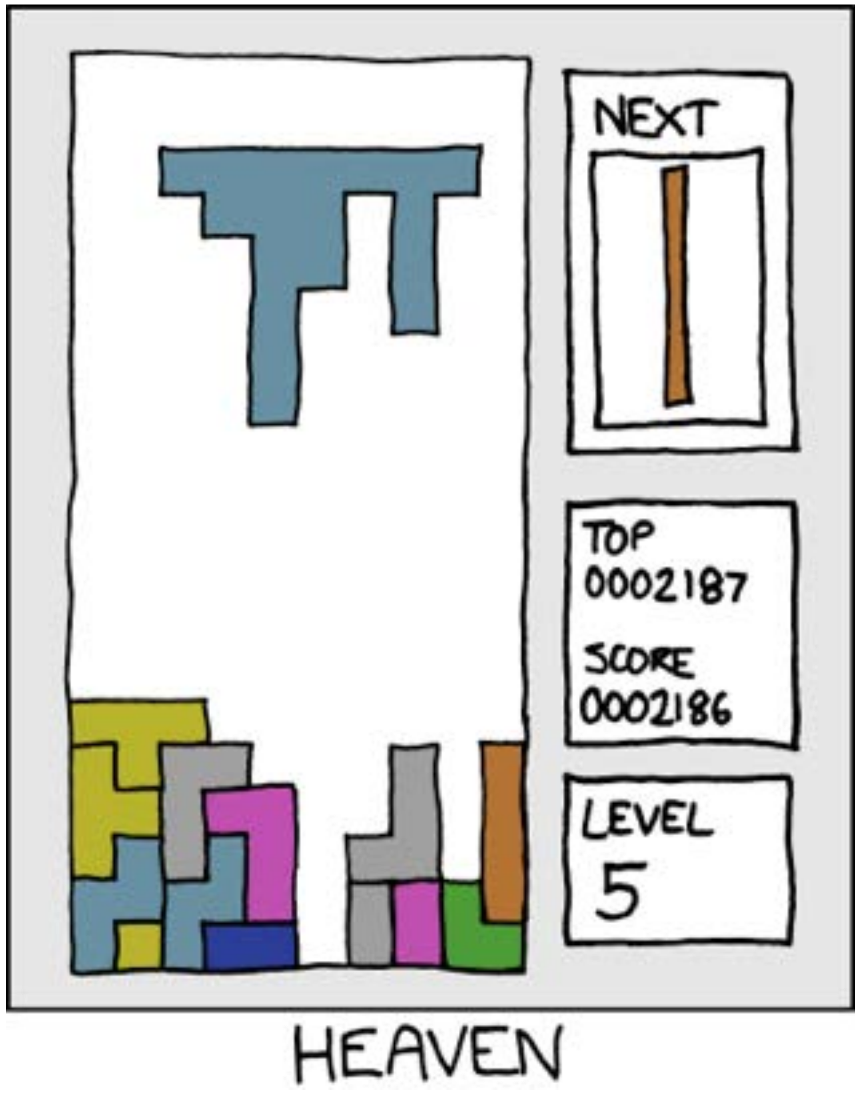
No PotW on Mon, Dec 7, 2015.

Participation is optional.
Prerequisite: Being registered for Algolab exam.
You have to register on the forum until latest
Mon **Nov 23, 2015, 10:00 AM** Zurich time.

Computer activity (screen) is logged.

HOW TO SOLVE PROBLEMS

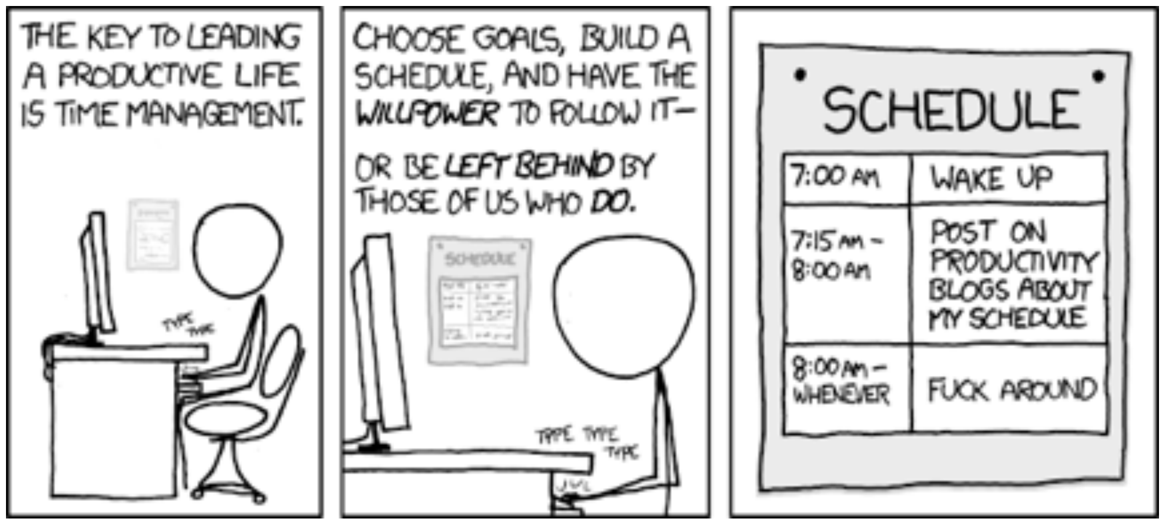
- Know what to know
- Understand your task
- Find an appropriate model
- Design an efficient algorithm
- Implement that algorithm
- Avoid "stupid" mistakes



http://xkcd.com/888/

KNOW WHAT TO KNOW

- Both the material from the tutorials and the collection of problems form the contents of this course.
- Key concepts, techniques, and skills were covered in the tutorials and/or practiced in a problem.
- Also meta skills such as time management play a role here (practiced in PotWs).


THE KEY TO LEADING A PRODUCTIVE LIFE IS TIME MANAGEMENT.
CHOOSE GOALS, BUILD A SCHEDULE, AND HAVE THE WILLPOWER TO FOLLOW IT— OR BE LEFT BEHIND BY THOSE OF US WHO DO.
TYPE TYPE TYPE
SCHEDULE
7:00 AM | WAKE UP
7:15 AM - 8:00 AM | POST ON PRODUCTIVITY BLOGS ABOUT MY SCHEDULE
8:00 AM - WHENEVER | FUCK AROUND

http://xkcd.com/874/

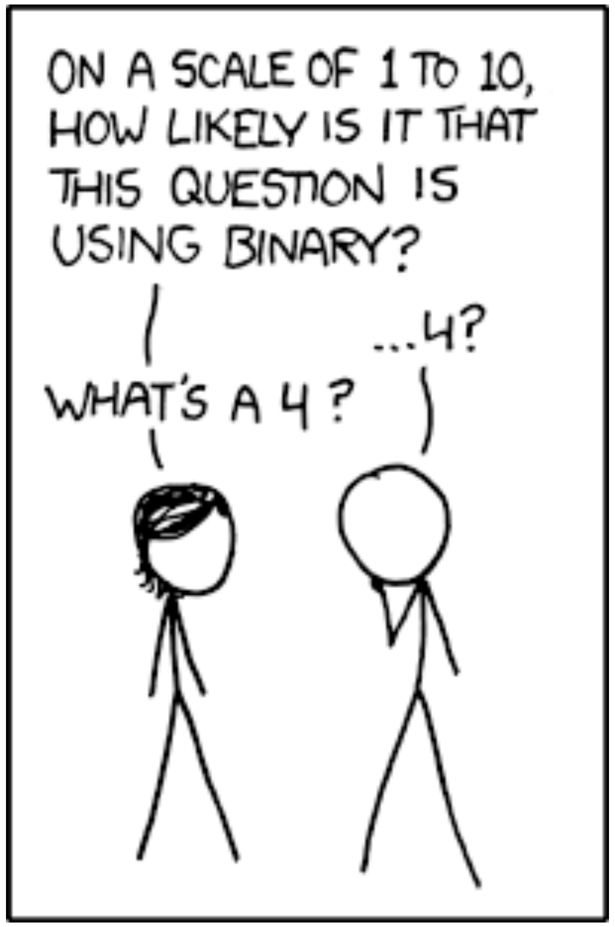
KNOW WHAT TO KNOW

- We will not ask you to do something drastically different from what you have seen during the semester.

  The problems from the exam preparation weeks give you a good idea of how problems in the exam may look like.

- If you use a data structure/algorithm/technique that was not covered, you are most likely not solving the problem in a way we intended.

  You go down a risky road. If that works out, kudos to you for the original approach! If not ... you knew the risks ...


ON A SCALE OF 1 TO 10, HOW LIKELY IS IT THAT THIS QUESTION IS USING BINARY?
WHAT'S A 4?
...4?

http://xkcd.com/953/

UNDERSTAND YOUR TASK

- Read the problem statement carefully.
- Read the problem statement again.

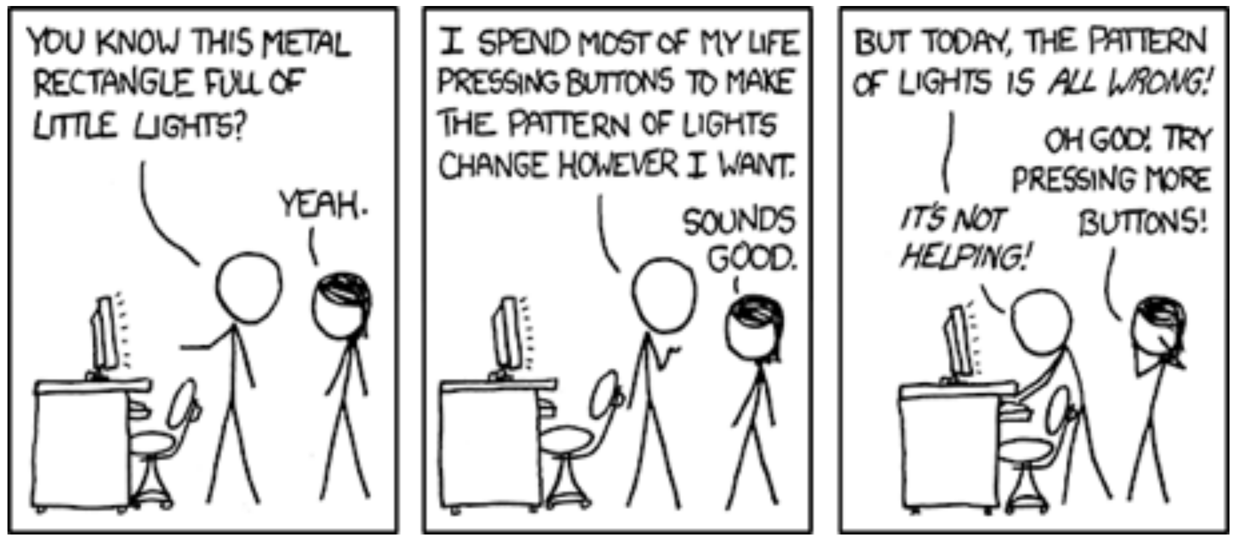
Make sure that you understand what is asked. Do not make any assumptions/ interpretations that are not clearly supported by what is written.

- Check the provided example(s) and if they concur with your understanding.

These examples are part of the problem description.

- If (and only if) you think the problem is not clearly stated, ask an assistant.

The assistants are not there to confirm your understanding. They will answer: ``Please read the problem statement. It is clearly stated.'' - unless they agree it is not clear.


YOU KNOW THIS METAL RECTANGLE FULL OF LITTLE LIGHTS?
YEAH.
I SPEND MOST OF MY LIFE PRESSING BUTTONS TO MAKE THE PATTERN OF LIGHTS CHANGE HOWEVER I WANT.
SOUNDS GOOD.
BUT TODAY, THE PATTERN OF LIGHTS IS ALL WRONG!
OH GOD! TRY PRESSING MORE BUTTONS!
IT'S NOT HELPING!

http://xkcd.com/722/

FIND A MODEL

Rephrase the problem in abstract/mathematical terms.

- (using terms like graph, vertex, edge, component, matching, point, line, matrix, relation, inequality, ...) rather than planes, aliens, countries, or antennas.
- Sometimes this task is straightforward and sometimes there are choices to make.
- The goal is to get rid of the story and unveil the algorithmic problem.

Body of Knowledge (Jaume Plensa, 2010)

ALGORITHM DESIGN

- How can you attack this problem?
- Do not get caught in the story!

You should let yourself get inspired by what you already know. Not on the story-level, but on the algorithmic level instead!

- Try to think about different alternatives: evaluate them briefly, which look promising?

  (LP, network flow, maximum matching, dynamic programming, Delaunay/Voronoi, minimum enclosing shapes, greedy, scan, binary search, shortest paths,...)
- If you have no idea, consider another problem.
- If your try gets stuck, consider a different approach.

  Performing such meta-searching effectively is an objective of this course.

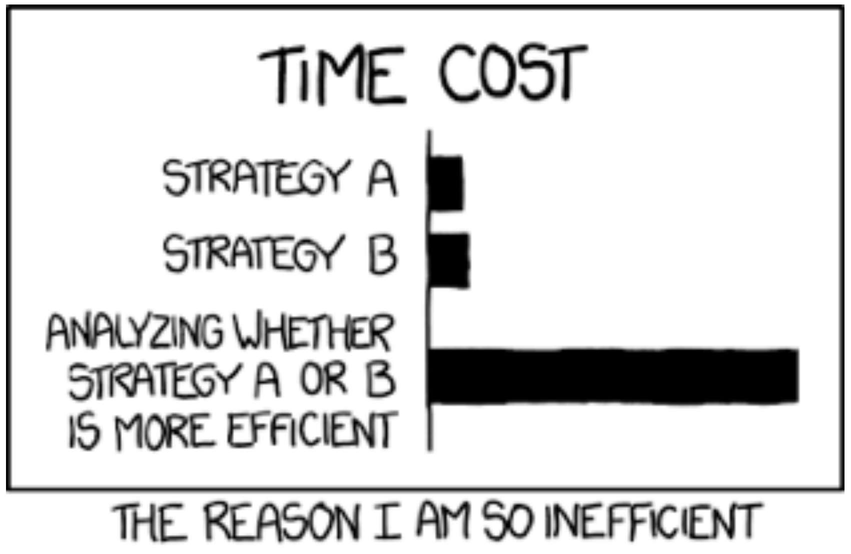
IMPLEMENTATION

- Every problem can be solved with no more than ~100 lines of well-written code.
- Use suitable data types for input/output processing (precision vs. speed).

Unlike for some earlier problems we will not tell you which type to use, because you were taught all the necessary bits...

- Avoid premature optimizations.
- Practice helps a lot...

That's why this is a lab. The more you practice, the less likely it is that you run into a particular issue for the first time during the exam...

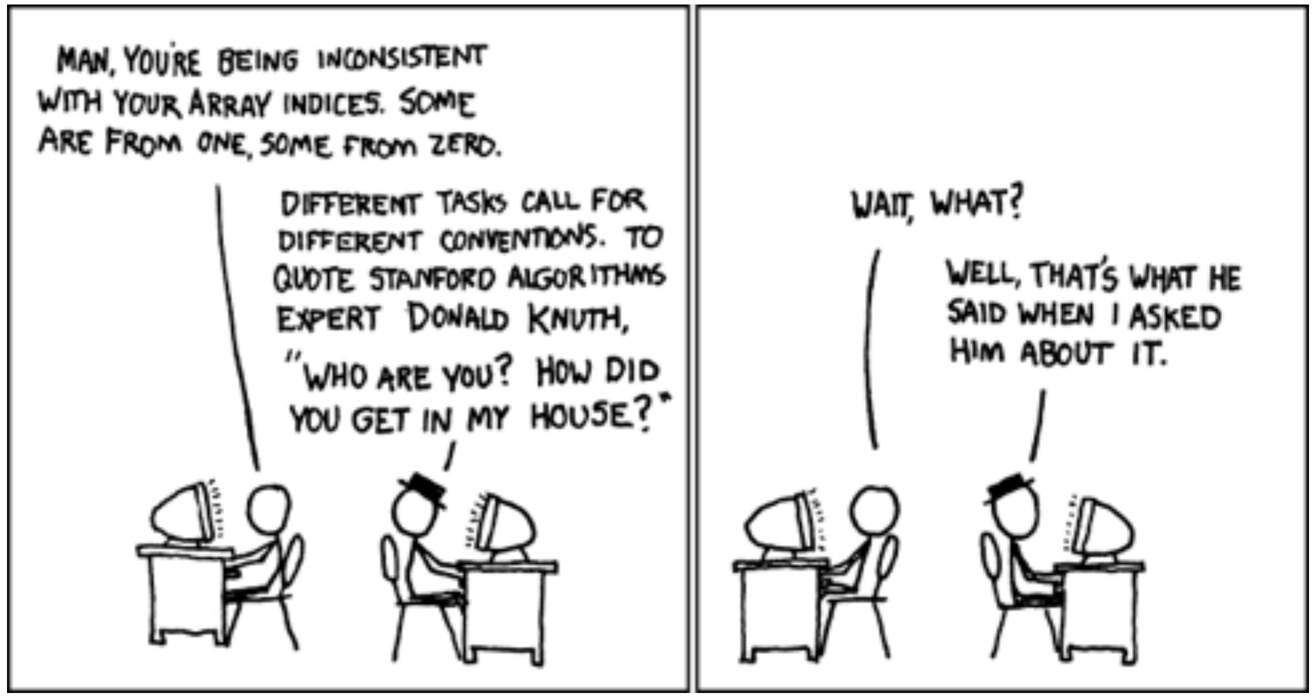

TIME COST
STRATEGY A
STRATEGY B
ANALYZING WHETHER STRATEGY A OR B IS MORE EFFICIENT
THE REASON I AM SO INEFFICIENT

http://xkcd.com/1445/

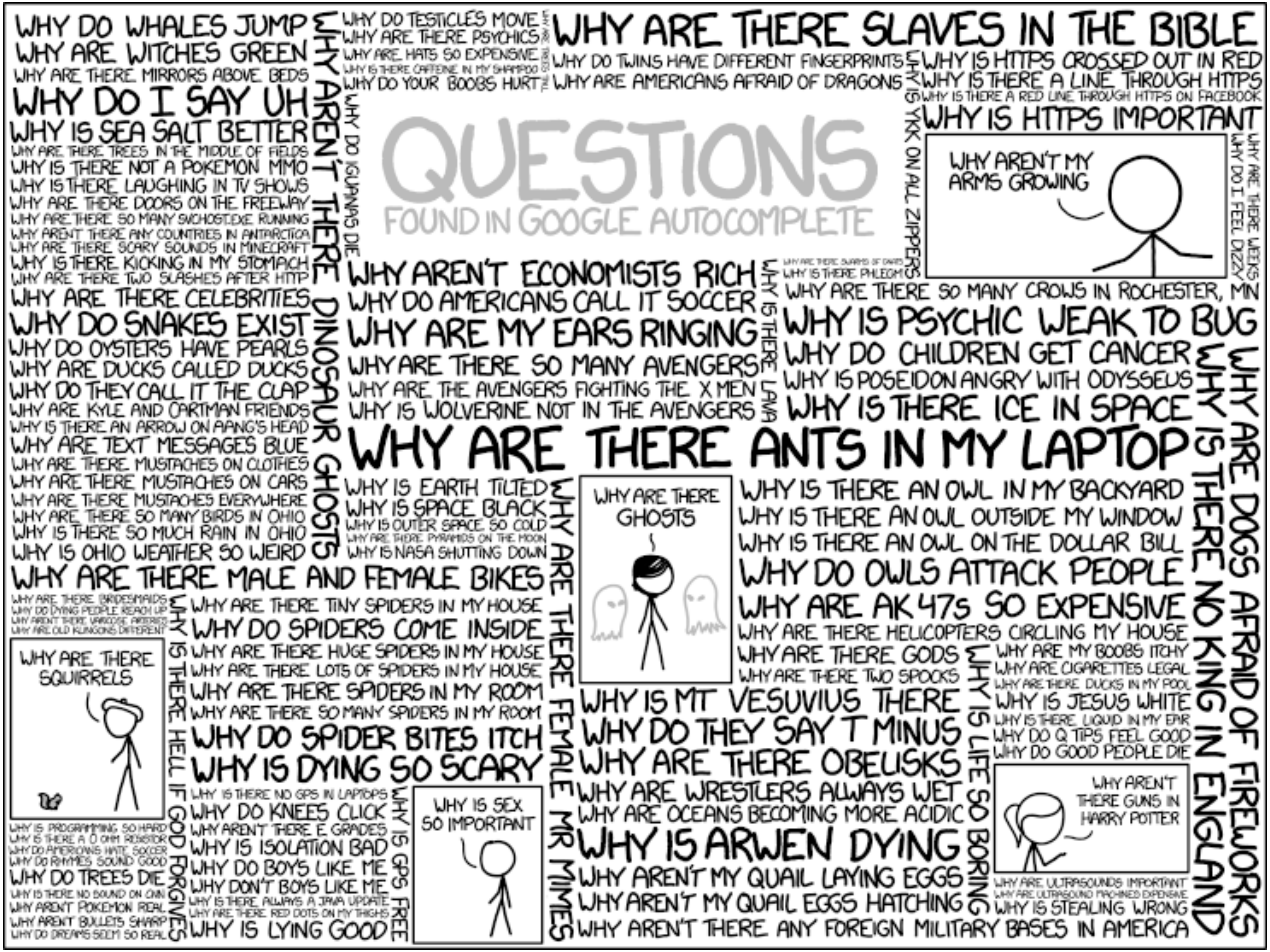
AVOID "STUPID" MISTAKES

- Submit to the right problem (correct link).
- Read all input, even if the result is determined half the way along already. Otherwise, you mess up a possibly following problem instance.
- `std::ios_base::sync_with_stdio(false);`
- `TMMTL ...`

  practice helps...


MAN, YOU'RE BEING INCONSISTENT WITH YOUR ARRAY INDICES. SOME ARE FROM ONE, SOME FROM ZERO.
DIFFERENT TASKS CALL FOR DIFFERENT CONVENTIONS. TO QUOTE STANFORD ALGORITHMS EXPERT DONALD KNUTH, "WHO ARE YOU? HOW DID YOU GET IN MY HOUSE?"
WAIT, WHAT?
WELL, THAT'S WHAT HE SAID WHEN I ASKED HIM ABOUT IT.

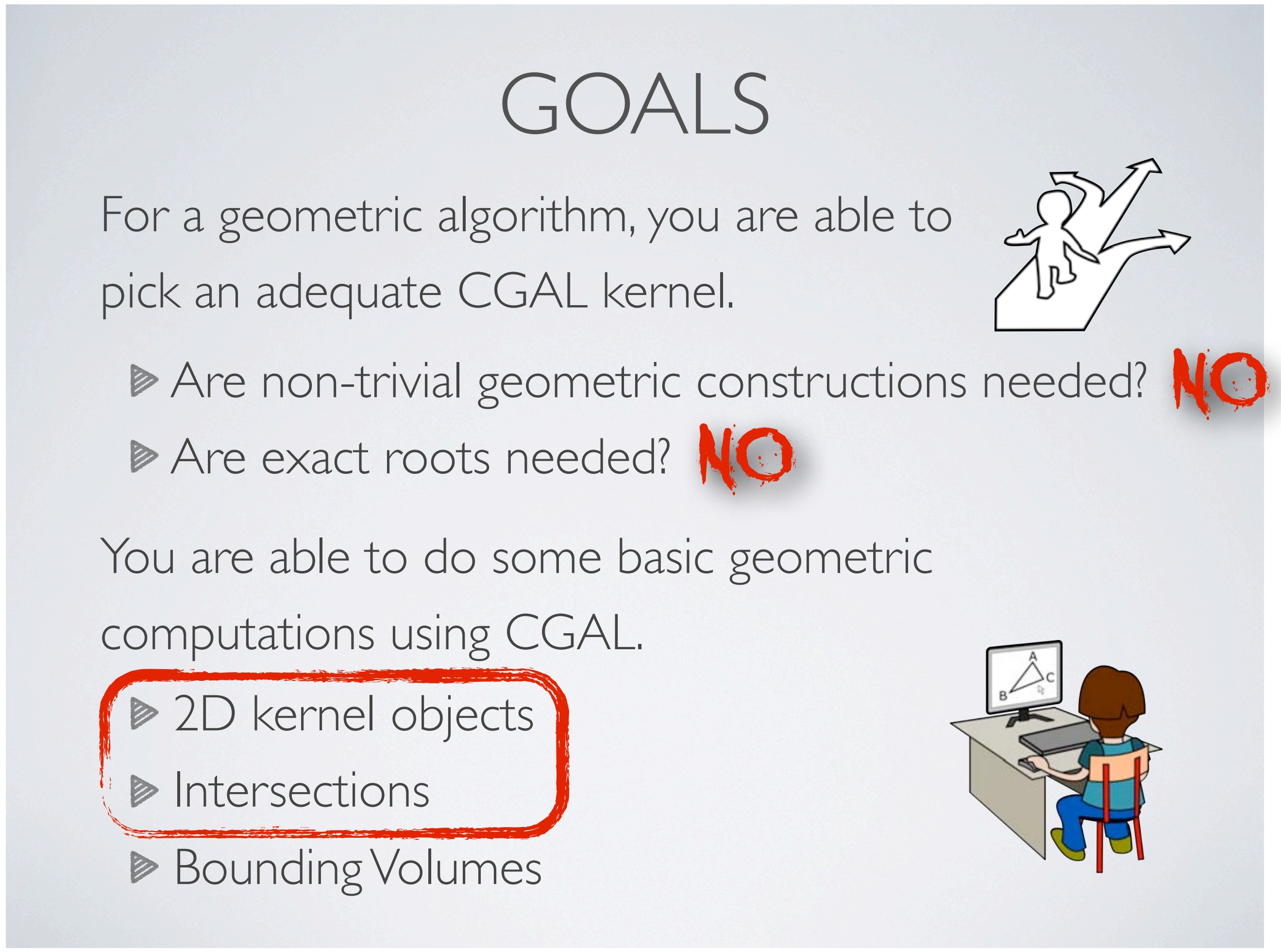
http://xkcd.com/1445/


QUESTIONS
FOUND IN GOOGLE AUTOCOMPLETE
WHY DO WHALES JUMP
WHY ARE WITCHES GREEN
WHY ARE THERE MIRRORS ABOVE BEDS
WHY DO I SAY UH
WHY IS SEA SALT BETTER
WHY ARE THERE TREES IN THE MIDDLE OF FIELDS
WHY IS THERE NOT A POKEMON MMO
WHY IS THERE LAUGHING IN TV SHOWS
WHY ARE THERE DOORS ON THE FREEWAY
WHY ARE THERE SO MANY SVCHOST.EXE RUNNING
WHY AREN'T THERE ANY COUNTRIES IN ANTARCTICA
WHY ARE THERE SCARY SOUNDS IN MINECRAFT
WHY IS THERE KICKING IN MY STOMACH
WHY ARE THERE TWO SLASHES AFTER HTTP
WHY ARE THERE CELEBRITIES
WHY DO SNAKES EXIST
WHY DO OYSTERS HAVE PEARLS
WHY ARE DUCKS CALLED DUCKS
WHY DO THEY CALL IT THE CLAP
WHY ARE KYLE AND CARTMAN FRIENDS
WHY IS THERE AN ARROW ON AANG'S HEAD
WHY ARE TEXT MESSAGES BLUE
WHY ARE THERE MUSTACHES ON CLOTHES
WHY ARE THERE MUSTACHES ON CARS
WHY ARE THERE MUSTACHES EVERYWHERE
WHY ARE THERE SO MANY BIRDS IN OHIO
WHY IS THERE SO MUCH RAIN IN OHIO
WHY IS OHIO WEATHER SO WEIRD
WHY AREN'T THERE DINOSAUR GHOSTS
WHY DO TESTICLES MOVE
WHY ARE THERE PSYCHICS
WHY ARE HATS SO EXPENSIVE
WHY IS THERE CAFFEINE IN MY SHAMPOO
WHY DO YOUR BOOBS HURT
WHY ARE TREES TALL
WHY DO IGUANAS DIE
WHY ARE THERE SLAVES IN THE BIBLE
WHY DO TWINS HAVE DIFFERENT FINGERPRINTS
WHY ARE AMERICANS AFRAID OF DRAGONS
WHY IS YKK ON ALL ZIPPERS
WHY IS HTTPS CROSSED OUT IN RED
WHY IS THERE A LINE THROUGH HTTPS
WHY IS THERE A RED LINE THROUGH HTTPS ON FACEBOOK
WHY IS HTTPS IMPORTANT
WHY AREN'T MY ARMS GROWING
WHY ARE THERE WEEKS
WHY DO I FEEL DIZZY
WHY AREN'T ECONOMISTS RICH
WHY DO AMERICANS CALL IT SOCCER
WHY ARE MY EARS RINGING
WHY ARE THERE SO MANY AVENGERS
WHY ARE THE AVENGERS FIGHTING THE X MEN
WHY IS WOLVERINE NOT IN THE AVENGERS
WHY IS THERE LAVA
WHY ARE THERE SWARMS OF GNATS
WHY IS THERE PHLEGM
WHY ARE THERE SO MANY CROWS IN ROCHESTER, MN
WHY IS PSYCHIC WEAK TO BUG
WHY DO CHILDREN GET CANCER
WHY IS POSEIDON ANGRY WITH ODYSSEUS
WHY IS THERE ICE IN SPACE
WHY ARE THERE ANTS IN MY LAPTOP
WHY IS EARTH TILTED
WHY IS SPACE BLACK
WHY IS OUTER SPACE SO COLD
WHY ARE THERE PYRAMIDS ON THE MOON
WHY IS NASA SHUTTING DOWN
WHY ARE THERE MALE AND FEMALE BIKES
WHY ARE THERE FEMALE MR MIMES
WHY ARE THERE GHOSTS
WHY IS THERE AN OWL IN MY BACKYARD
WHY IS THERE AN OWL OUTSIDE MY WINDOW
WHY IS THERE AN OWL ON THE DOLLAR BILL
WHY DO OWLS ATTACK PEOPLE
WHY ARE AK 47s SO EXPENSIVE
WHY ARE THERE HELICOPTERS CIRCLING MY HOUSE
WHY ARE THERE GODS
WHY ARE THERE TWO SPOCKS
WHY IS THERE NO KING IN ENGLAND
WHY ARE DOGS AFRAID OF FIREWORKS
WHY ARE THERE BRIDESMAIDS
WHY DO DYING PEOPLE REACH UP
WHY AREN'T THERE VARICOSE ARTERIES
WHY ARE OLD KLINGONS DIFFERENT
WHY ARE THERE SQUIRRELS
WHY IS THERE HELL IF GOD FORGIVES
WHY ARE THERE TINY SPIDERS IN MY HOUSE
WHY DO SPIDERS COME INSIDE
WHY ARE THERE HUGE SPIDERS IN MY HOUSE
WHY ARE THERE LOTS OF SPIDERS IN MY HOUSE
WHY ARE THERE SPIDERS IN MY ROOM
WHY ARE THERE SO MANY SPIDERS IN MY ROOM
WHY DO SPIDER BITES ITCH
WHY IS DYING SO SCARY
WHY IS PROGRAMMING SO HARD
WHY IS THERE A 0 OHM RESISTOR
WHY DO AMERICANS HATE SOCCER
WHY DO RHYMES SOUND GOOD
WHY DO TREES DIE
WHY IS THERE NO SOUND ON CNN
WHY AREN'T POKEMON REAL
WHY AREN'T BULLETS SHARP
WHY DO DREAMS SEEM SO REAL
WHY IS THERE NO GPS IN LAPTOPS
WHY DO KNEES CLICK
WHY AREN'T THERE E GRADES
WHY IS ISOLATION BAD
WHY DO BOYS LIKE ME
WHY DON'T BOYS LIKE ME
WHY IS THERE ALWAYS A JAVA UPDATE
WHY ARE THERE RED DOTS ON MY THIGHS
WHY IS LYING GOOD
WHY IS GPS FREE
WHY IS SEX SO IMPORTANT
WHY IS MT VESUVIUS THERE
WHY DO THEY SAY T MINUS
WHY ARE THERE OBELISKS
WHY ARE WRESTLERS ALWAYS WET
WHY ARE OCEANS BECOMING MORE ACIDIC
WHY IS ARWEN DYING
WHY AREN'T MY QUAIL LAYING EGGS
WHY AREN'T MY QUAIL EGGS HATCHING
WHY AREN'T THERE ANY FOREIGN MILITARY BASES IN AMERICA
WHY IS LIFE SO BORING
WHY ARE MY BOOBS ITCHY
WHY ARE CIGARETTES LEGAL
WHY ARE THERE DUCKS IN MY POOL
WHY IS JESUS WHITE
WHY IS THERE LIQUID IN MY EAR
WHY DO Q TIPS FEEL GOOD
WHY DO GOOD PEOPLE DIE
WHY AREN'T THERE GUNS IN HARRY POTTER
WHY ARE ULTRASOUNDS IMPORTANT
WHY ARE ULTRASOUND MACHINES EXPENSIVE
WHY IS STEALING WRONG

http://xkcd.com/1256/

HIT

Problem: Given a ray and some line segments, does the ray intersect any segment?



GOALS

For a geometric algorithm, you are able to pick an adequate CGAL kernel.

- Are non-trivial geometric constructions needed? NO
- Are exact roots needed? NO

You are able to do some basic geometric computations using CGAL.

- 2D kernel objects
- Intersections
- Bounding Volumes

Do not just guess, make a conscious decision!

Of course, you can just try all three kernels.

But often this choice also impacts the design of the algorithm.

No non-trivial constructions needed

$\rightarrow$ `CGAL::Exact_predicates_inexact_constructions_kernel`

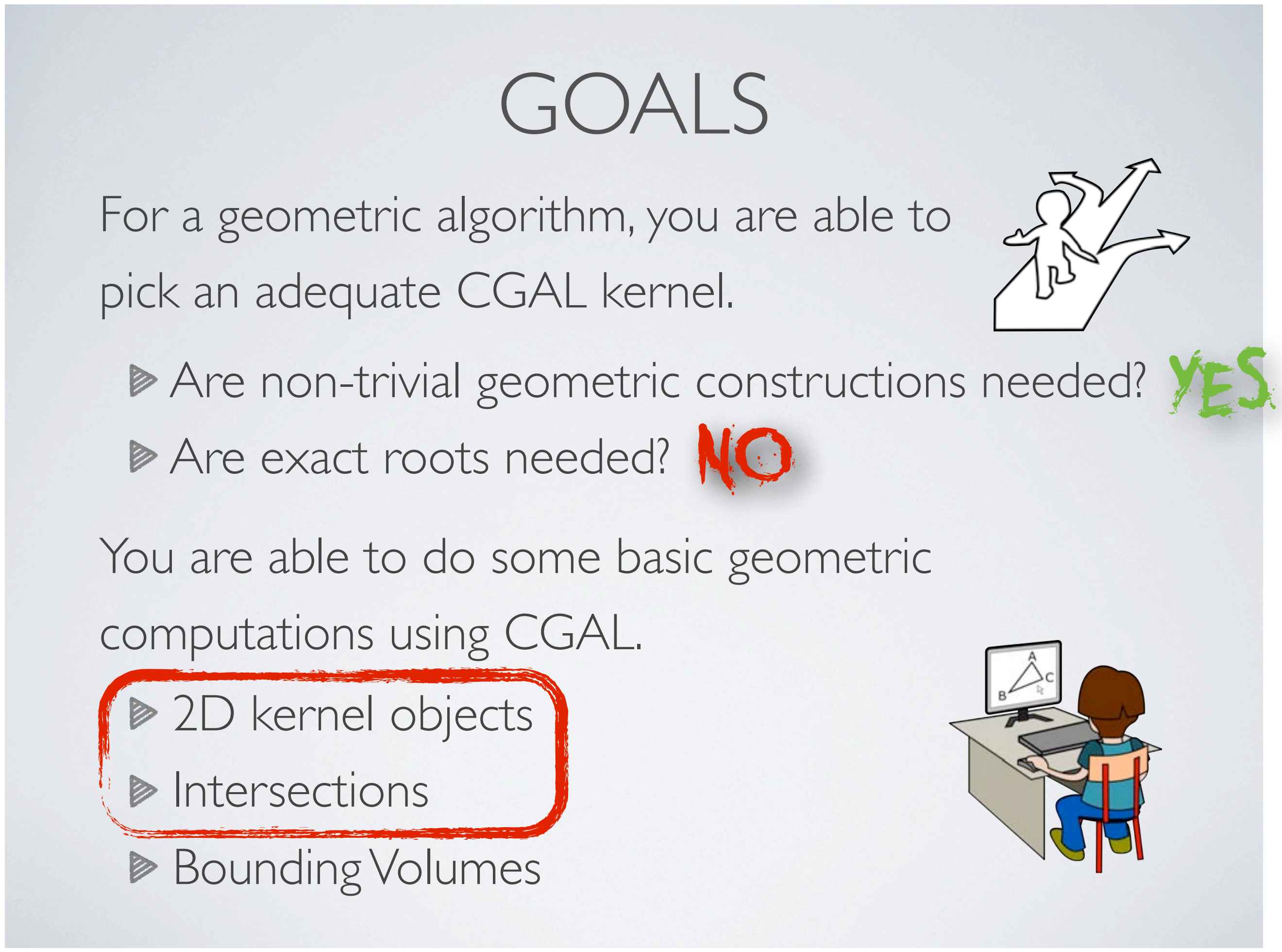
HIT

```
#include <CGAL/Exact_predicates_inexact_constructions_kernel.h>
#include <iostream>

typedef CGAL::Exact_predicates_inexact_constructions_kernel K;

int main()
{
  std::ios_base::sync_with_stdio(false);
  for (std::size_t n; std::cin >> n && n > 0;) {
    K::Ray_2 r;
    std::cin >> r;
    bool found = false;
    do {
      K::Segment_2 s;
      std::cin >> s;
      if (!found && CGAL::do_intersect(s,r)) found = true;
    } while (--n > 0);
    std::cout << (found ? "yes" : "no") << std::endl;
  }
}
```

With 52-bit numbers (–> double) and EPIC-kernel this is fine here.
In general, be careful (–> IO performance slide from CGAL tutorial I)...

FIRSTHIT

Problem: Given a ray and some line segments, where does the ray first intersect a segment?



GOALS

For a geometric algorithm, you are able to pick an adequate CGAL kernel.

- Are non-trivial geometric constructions needed? YES
- Are exact roots needed? NO

You are able to do some basic geometric computations using CGAL.

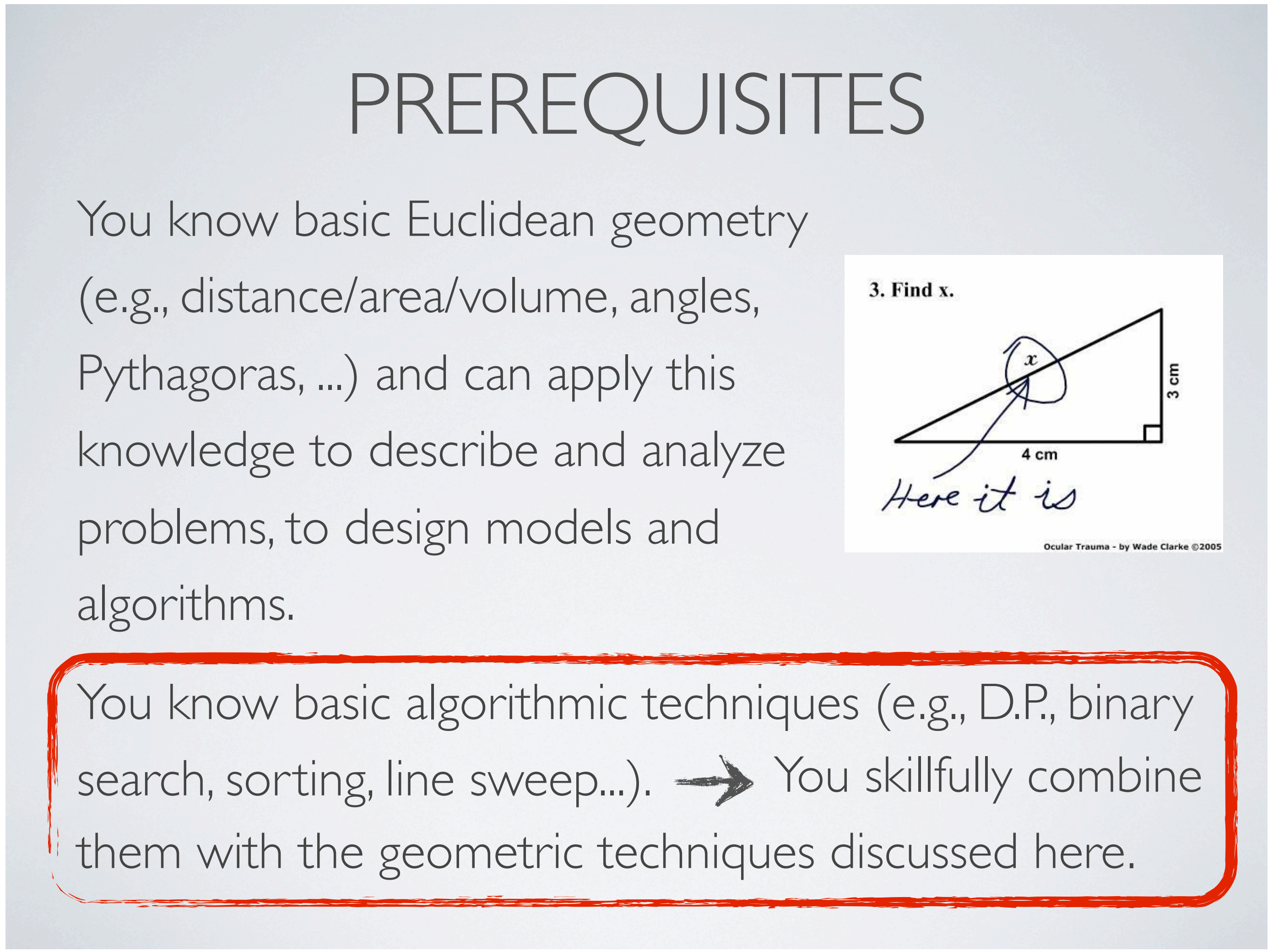
- 2D kernel objects
- Intersections
- Bounding Volumes

Need intersection point

→ `CGAL::Exact_predicates_exact_constructions_kernel`

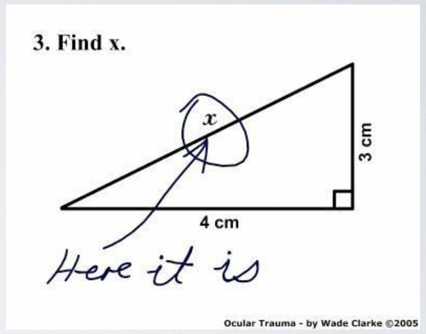
FIRSTHIT

Problem: Given a ray and some line segments, where does the ray first intersect a segment?



PREREQUISITES

You know basic Euclidean geometry (e.g., distance/area/volume, angles, Pythagoras, ...) and can apply this knowledge to describe and analyze problems, to design models and algorithms.



You know basic algorithmic techniques (e.g., D.P., binary search, sorting, line sweep...). $\rightarrow$ You skillfully combine them with the geometric techniques discussed here.

Most expensive operation is intersection construction

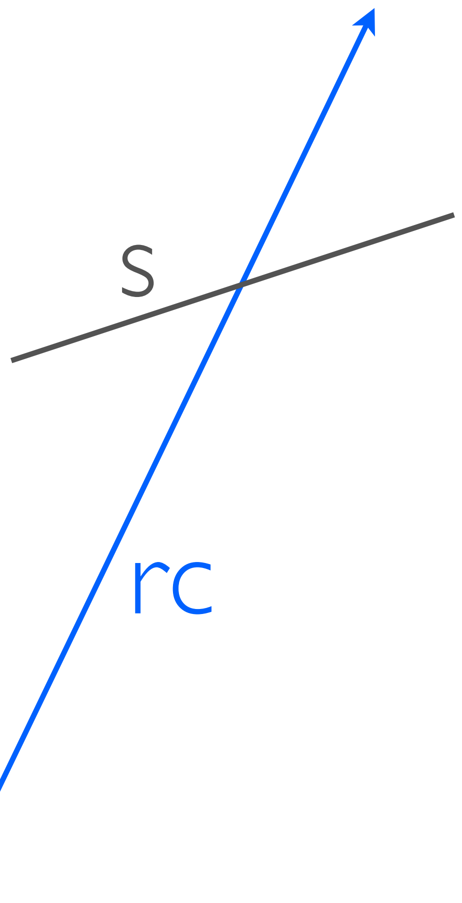
$\rightarrow$ aim to minimize the number of constructions

FIRSTHIT

Problem: Given a ray and some line segments, where does the ray first intersect a segment?

aim to minimize the number of constructions How?

- Find a segment s that intersects the ray r.
  If no such segment exists, we are done.
- Clip r at s∩r to a segment rc.
- Continue to test the remaining segments against rc; clip rc at each intersection.



→ no intersection construction for segments that intersect r but not rc

FIRSTHIT

Problem: Given a ray and some line segments, where does the ray first intersect a segment?

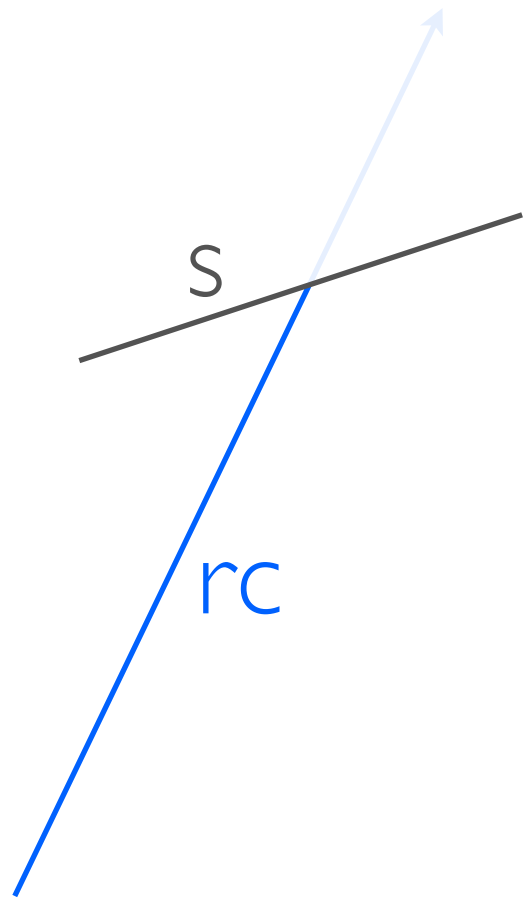
no intersection construction for segments that intersect r but not rc

What's the gain?

If we find the first segment hit by r early, no further intersection is constructed.

→ randomize segment order.

Claim. Only O(log n) intersections constructed in expectation.



FIRSTHIT

Consider a permutation $x = x_1,\dots,x_n$ of $\{1,\dots, n\}$ u.a.r. and the sequence $m = m_0, m_1,\dots, m_n$ with $m_i = \min \{x_1,\dots, x_i\}$. How many changes occur in this sequence? $m0 = -\infty$

(indices $i \in \{1,\dots, n\}$ with $m_i \neq m_{i-1}$)

Example. $n = 7$ and $x = 4,2,6,1,7,3,5$

$m = 4,2,2,1,1,1,1$ with 3 changes

Expected #changes $= E(\Sigma C_i) = \Sigma E(C_i)$ $\quad C_i = \begin{cases} 1, \text{ if } m_i \neq m_{i-1} \\ 0, \text{ otherwise} \end{cases}$

$$= \overset{m_1}{\mathbf{1}} + \overset{m_2}{\mathbf{\tfrac{1}{2}}} + \overset{m_3}{\mathbf{\tfrac{1}{3}}} + \dots + \overset{m_n}{\mathbf{\tfrac{1}{n}}} = \overset{\text{Harmonic number}}{H_n} = \Theta(\log n)$$

Same as Firsthit with segment shortening
(intersection is only constructed if new segment is hit before) $\longrightarrow$ Claim

FIRSTHIT

```
#include <CGAL/Exact_predicates_exact_constructions_kernel.h>
#include <vector>
#include <algorithm>
#include <stdexcept>

typedef CGAL::Exact_predicates_exact_constructions_kernel K;

...

void find_hit(std::size_t n) {
  // read input
  K::Ray_2 r;
  std::cin >> r;
  std::vector<K::Segment_2> segs;
  segs.reserve(n);
  double ix, iy, jx, jy;
  for (std::size_t i = 0; i < n; ++i) {
    // as each coordinate can be represented as double, this is
    // significantly faster than K::Segment_2 s; std::cin >> s;
    std::cin >> ix >> iy >> jx >> jy;
    segs.push_back(K::Segment_2(K::Point_2(ix,iy),K::Point_2(jx,jy)));
  }
  std::random_shuffle(segs.begin(), segs.end());

  ...
}

int main()
{
  std::cin.sync_with_stdio(false);
  std::cout << std::setiosflags(std::ios::fixed) << std::setprecision(0);
  for (std::size_t n; std::cin >> n && n > 0;)
    find_hit(n);
  return 0;
}
```

important to avoid worst case orders

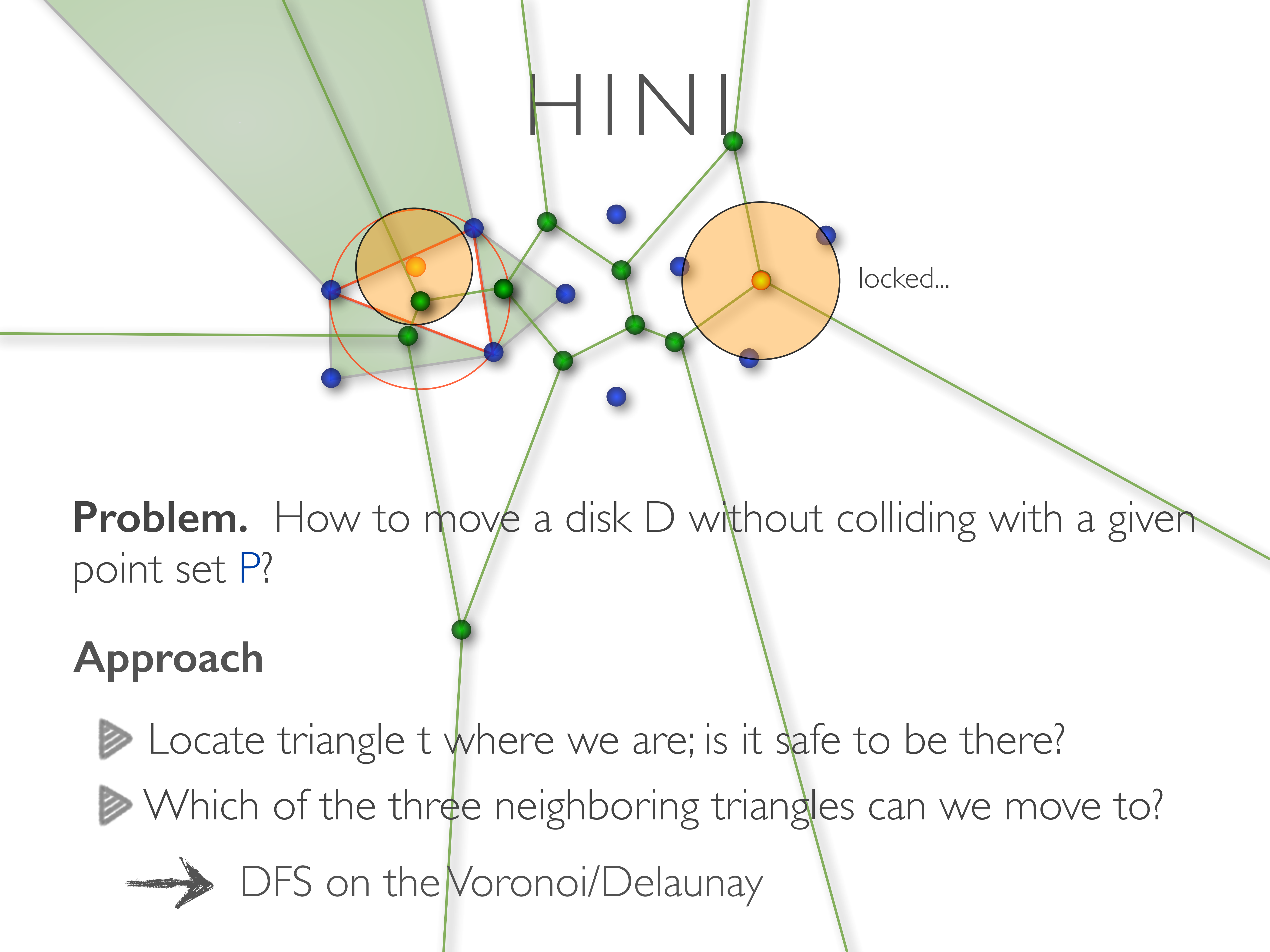
FIRSTHIT

```
// find some segment hit by r
K::Segment_2 rc(r.source(), r.source()); // clipped ray
std::size_t i = 0;
for (; i < n; ++i)
  if (CGAL::do_intersect(segs[i], r)) {
    shorten_segment(rc, CGAL::intersection(segs[i], r));
    break;
  }
if (i == n) { std::cout << "no\n"; return; }
// check remaining segments against rc
while (++i < n)
  if (CGAL::do_intersect(segs[i], rc))
    shorten_segment(rc, CGAL::intersection(segs[i], r)); // not rc!

std::cout << floor_to_double(rc.target().x()) << " "
          << floor_to_double(rc.target().y()) << "\n";
```

FIRSTHIT

```
typedef std::result_of<K::Intersect_2(K::Ray_2,K::Segment_2)>::type IT;


// clip/set target of s to o
void shorten_segment(K::Segment_2& s, const IT& o)
{
  if (const K::Point_2* p = boost::get<K::Point_2>(&*o))
    s = K::Segment_2(s.source(), *p);
  else if (const K::Segment_2* t = boost::get<K::Segment_2>(&*o))
    // select endpoint of *t closer to s.source()
    if (CGAL::collinear_are_ordered_along_line
         (s.source(), t->source(), t->target()))
      s = K::Segment_2(s.source(), t->source());
    else
      s = K::Segment_2(s.source(), t->target());
  else
    throw std::runtime_error("Strange segment intersection.");
}
```

HINT



Problem. How to move a disk D without colliding with a given point set P?

Approach

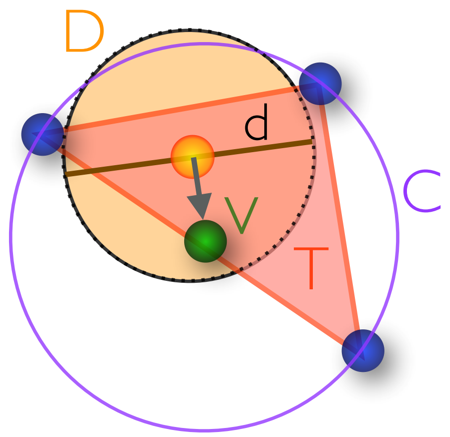
- Locate triangle t where we are; is it safe to be there?
- Which of the three neighboring triangles can we move to?

→ DFS on the Voronoi/Delaunay

HINI

Lemma. If the center of D is inside a triangle T of the DT of P and the placement is safe, then D can be safely moved straight s.t. its center coincides with the circumcenter of T.

- c ≔ center of D
- C ≔ circumdisk of T
- v ≔ center of C
- d ≔ diametrical segment of D orthogonal to vc.

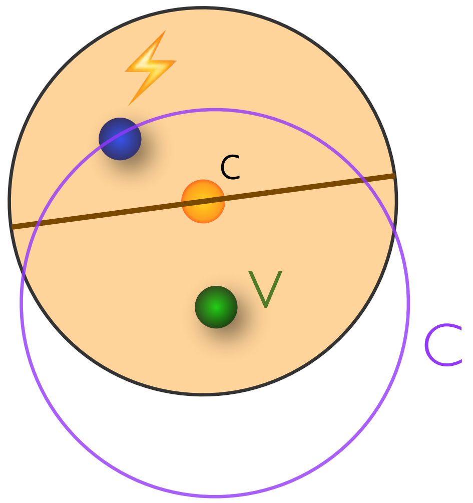


- Obvious if d lies inside C.

  The disk C is empty and the part of D outside C can only shrink (as a subset of $\mathbb{R}^2$) during the movement. As the radius of D is at most the radius of C, the final placement is safe.

- What if the endpoints of d are outside C?

  As the center of D is contained in T, there must be a vertex of T (which is a point from P) "behind" d (as seen from v). But this region lies inside D, in contradiction to D being empty of points from P.



HINI

Challenges

Use an external map where each access costs O(log n) time.
Or store it in a face directly to achieve constant access time.

- How to mark faces as visited to avoid cycling?
- These marks have to be initialized. If this is done explicitly for each query, it can be costly if there are many queries.

A query may escape or get stuck quickly, without having to search the whole graph...

Repetitive initialization can be avoided, e.g., using a counter "visited in query i" rather than a bool.

Obs. In the worst case, a query has to search its way through a linear number of faces.

Better: precompute for each triangle/Voronoi vertex the maximum escape radius (Dijkstra in O(n log n)).

HINI

```
typedef CGAL::Triangulation_vertex_base_2<K>              Vb;
typedef CGAL::Triangulation_face_base_with_info_2<int,K>    Fb;
typedef CGAL::Triangulation_data_structure_2<Vb,Fb>        Tds;
typedef CGAL::Delaunay_triangulation_2<K,Tds>              Delaunay;

try {
  if (r <= 0) throw true;
    Face_handle f = t.locate(s);
    if (CGAL::squared_distance(s, t.nearest_vertex(s, f)->point()) < r)
    throw false;
    // DFS
    std::vector<Face_handle> stack;
    stack.push_back(f);
    f->info() = i;
    while (!stack.empty()) {
    f = stack.back();
      stack.pop_back();
      if (t.is_infinite(f)) throw true;
      for (int j = 0; j < 3; ++j)
        if (f->neighbor(j)->info() < i &&
            t.segment(Delaunay::Edge(f,j)).squared_length() >= 4 * r) {
        stack.push_back(f->neighbor(j));
           f->neighbor(j)->info() = i;
        }
    }
    throw false;
}
catch (bool solvable)  { std::cout << (solvable ? "y" : "n"); }
```

s = query position, r = (squared) query radius

query number