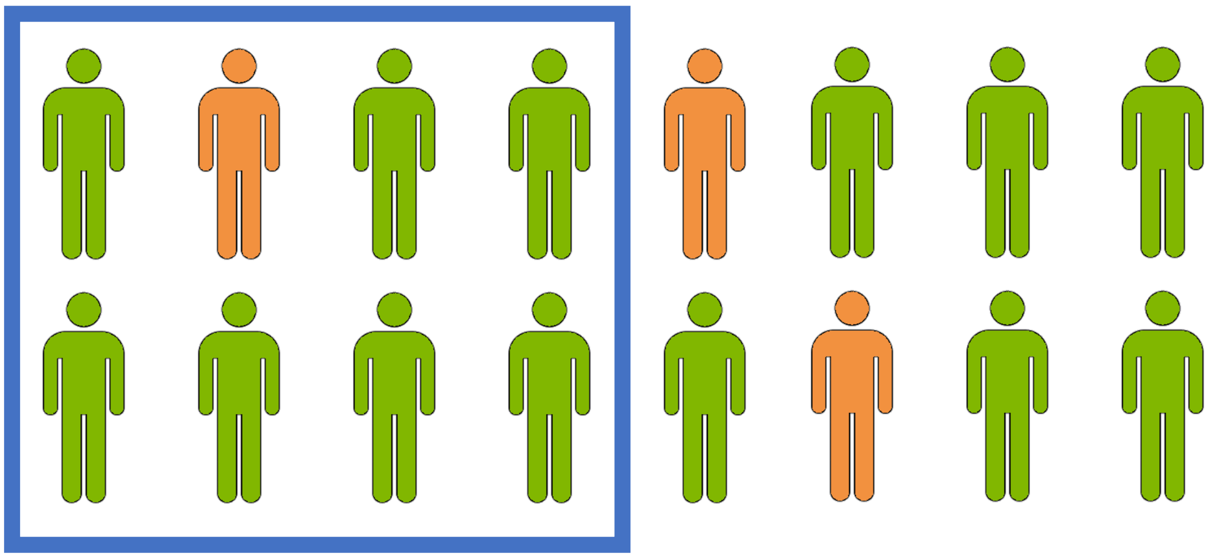
Inference Problems in the Linear Regime

Lessons from group testing



Matthew Aldridge
University of Leeds, UK

Workshop on Inference Problems: Algorithms and Lower Bounds
September 2020

1

Inference problems

Inference problems

There are a large number
n of inputs

Inferring all
their values takes
many (perhaps n)
measurements

Inference problems

There are a large number n of inputs

But only a small number k of the inputs are active

Inferring all their values takes many (perhaps n) measurements

Finding the active inputs and inferring their values takes few (perhaps $k \log n$) measurements

Inference problems

There are a large number n of inputs

But only a small number k of the inputs are active

Statistical models with n parameters:
Typically we need at least n pieces of data.
But if we know all but k of the parameters are zero,
we require less data.

Inference problems

There are a large number n of inputs

But only a small number k of the inputs are active

Statistical models with n parameters:
Typically we need at least n pieces of data.
But if we know all but k of the parameters are zero,
we require less data.

Compressed sensing:
Typically solving simultaneous linear equations in n variables
requires n equations,
but if the solution is k-sparse (in some basis) we require fewer.

Inference problems

There are a large number n of inputs

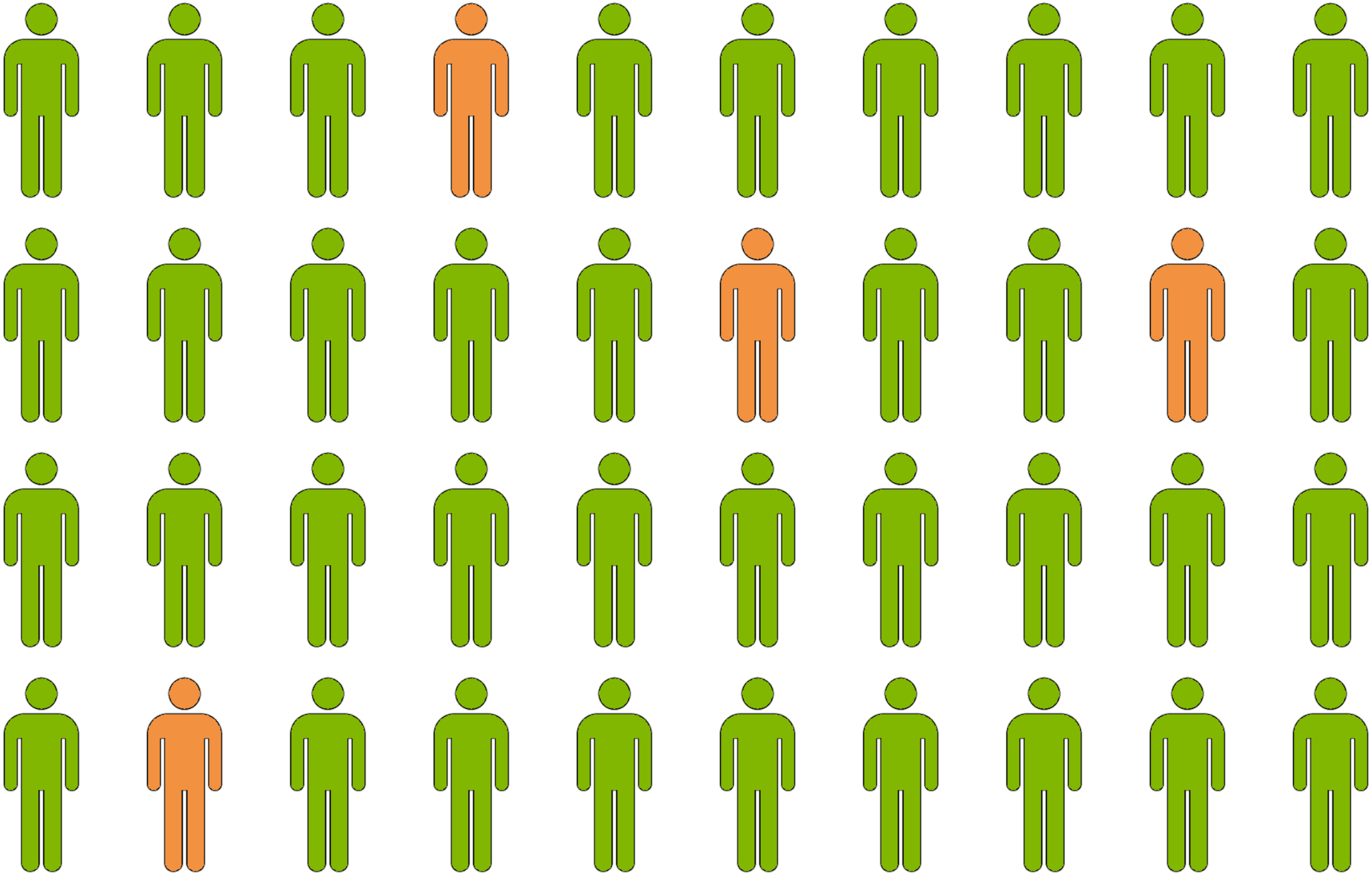
But only a small number k of the inputs are active

Statistical models with n parameters:
Typically we need at least n pieces of data.
But if we know all but k of the parameters are zero,
we require less data.

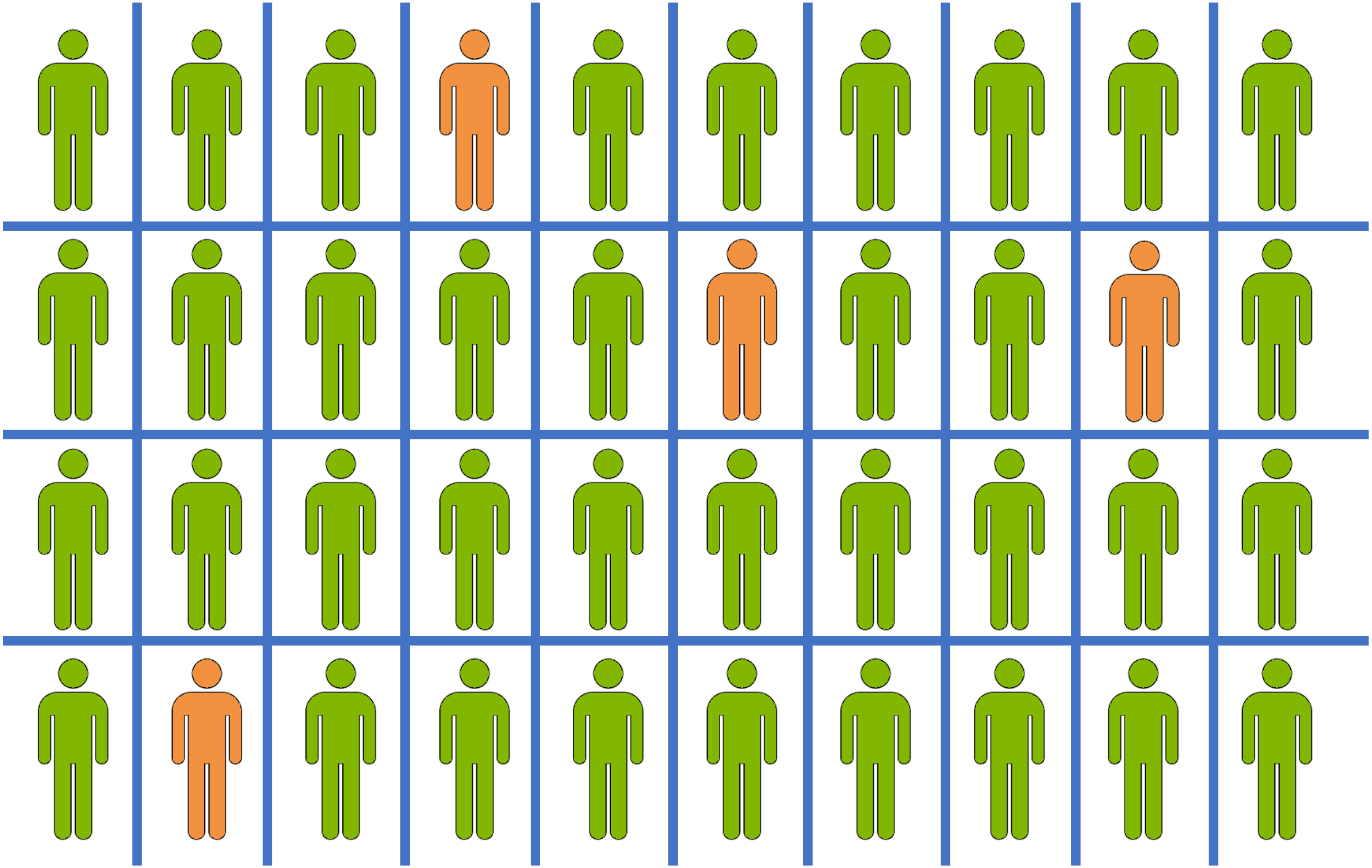
Compressed sensing:
Typically solving simultaneous linear equations in n variables requires n equations,
but if the solution is k-sparse (in some basis) we require fewer.

Pooled group testing…

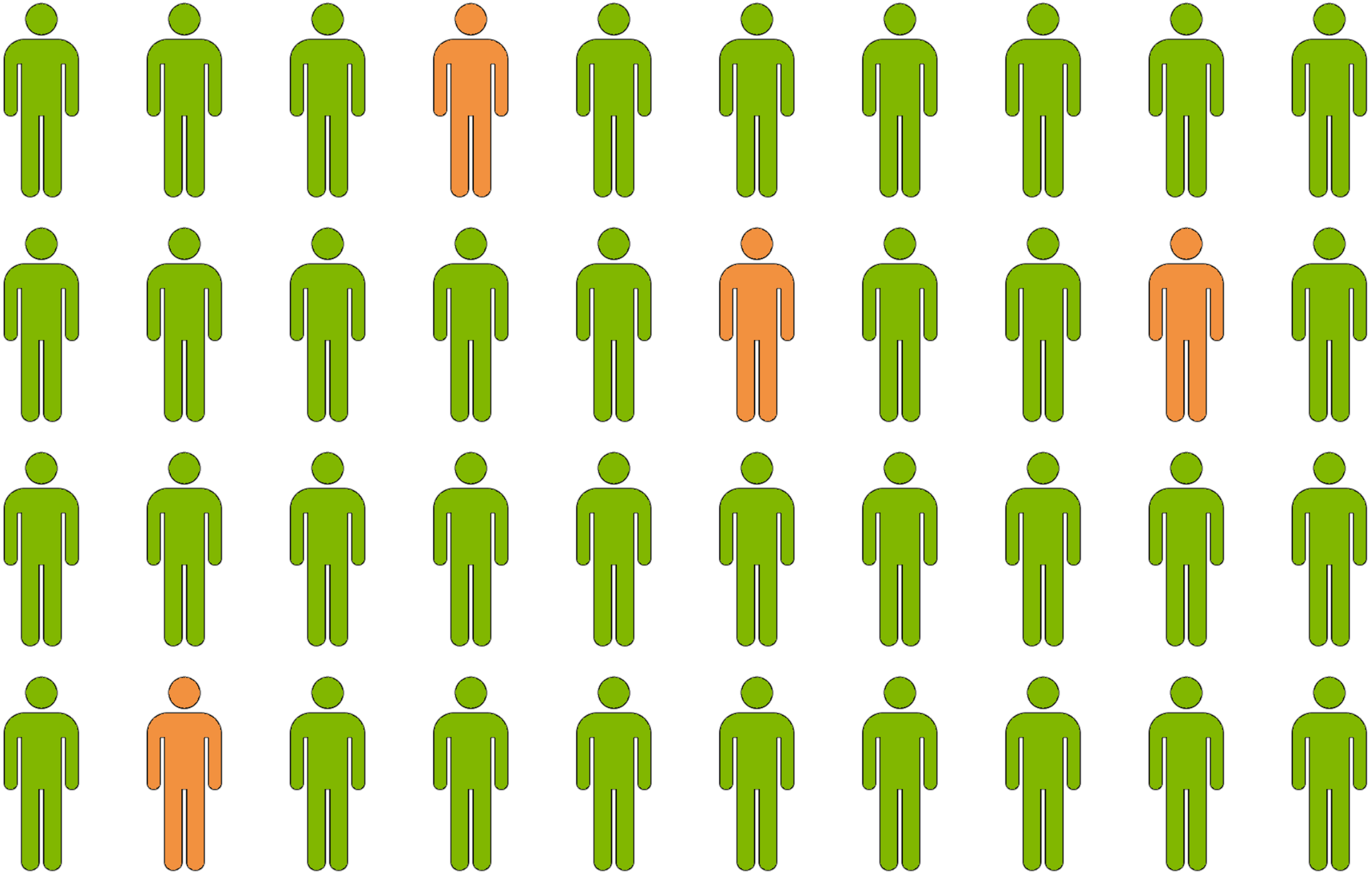
Group testing



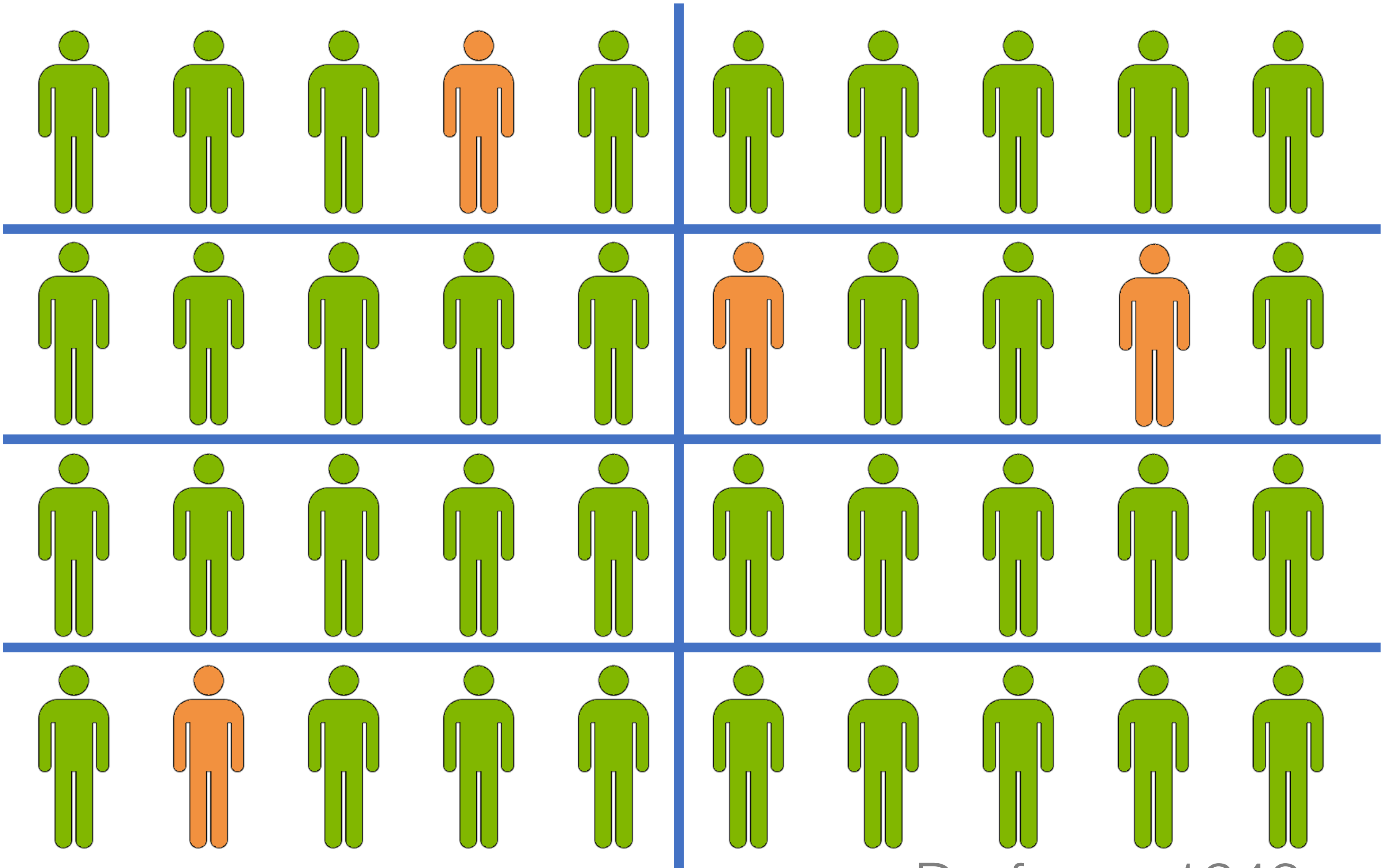
Group testing



Group testing

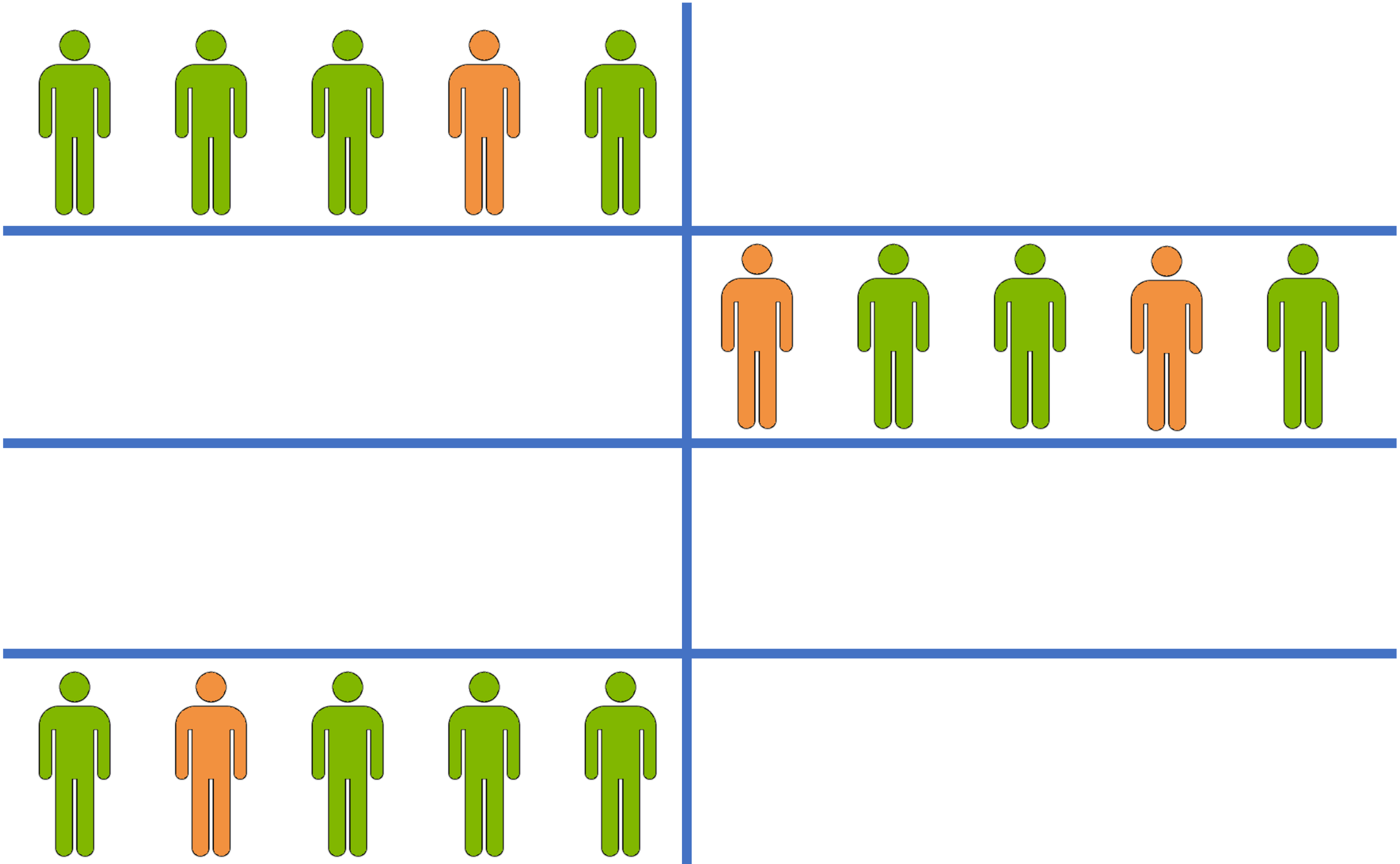


Group testing



Dorfman, 1943

Group testing



Dorfman, 1943

Types of problem

Adaptive
look at previous tests
before designing the next

Nonadaptive
all tests designed
in advance

Group testing

n items (soldiers)

k defective items (soldiers with syphilis)

T tests: *"Does this group of items contain at least one defective item?"* (blood tests)

Main problem

n items

k defective items

T tests

Given n and k,
how many tests T do we need
to reliably work out
which items were defective?

Main problem

n items

k defective items

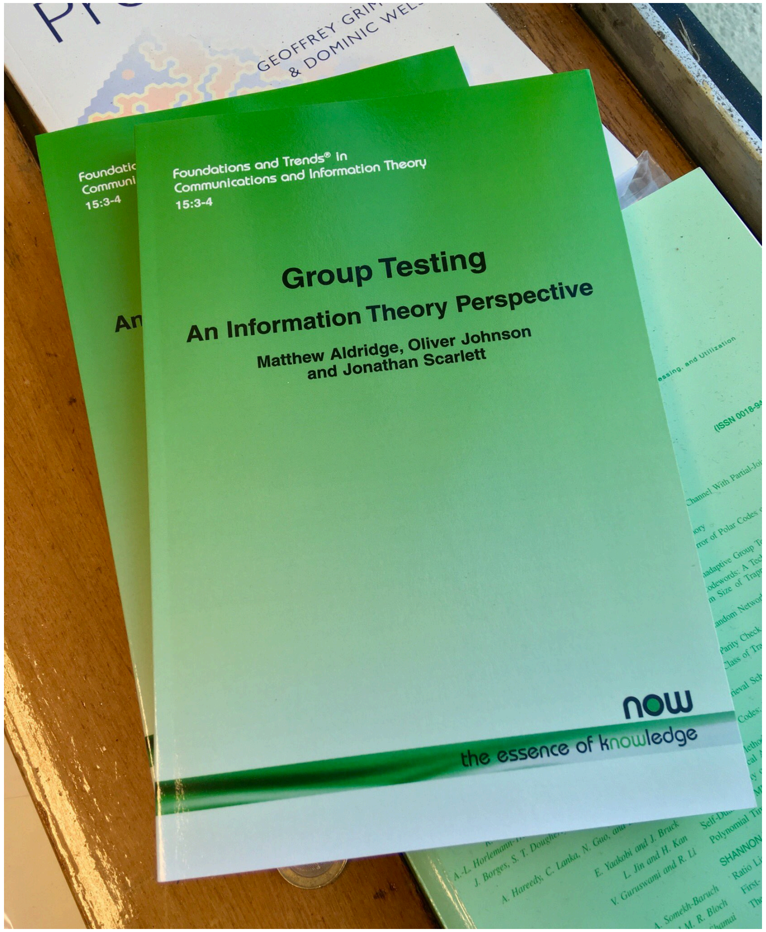
T tests

We can test individually with $T = n$ tests.

If k is small, can we manage with fewer?

M Aldridge, O Johnson and J Scarlett
Group Testing: An Information Theory Perspective
Foundations and Trends in Communications
and Information Theory, 2019

Preprint:
`arXiv:1902.06002`



Why should I care?

Why should I care?

Applications

Testing soldiers for syphilis

Testing for COVID-19 with limited test capacity

DNA screening

Management of wireless networks

Database management

Data compression

Cybersecurity

Graph learning

The counterfeit coin problem

…

Why should I care?

Applications

Concrete example of more general problems

Sparse inference, $p > n$ statistics

Nonlinear models

Search problems

Inverse problems

Why should I care?

Applications

Concrete example of more general problems

A fun problem in its own right

Probability

Statistics

Computer science

Information theory

Combinatorics

Types of group testing

Adaptive
Look at previous tests before designing the next

Nonadaptive
All tests designed in advance

Types of group testing

Adaptive
Look at previous tests
before designing the next

Nonadaptive
All tests designed
in advance

Types of group testing

Adaptive Look at previous tests before designing the next	**Nonadaptive** All tests designed in advance
Combinatorial Exactly *k* defective items Worst-case number of tests	**Probabilistic** Each item defective with prob *k*/*n* Typical number of tests

Types of group testing

Adaptive
Look at previous tests
before designing the next

Nonadaptive
All tests designed
in advance

Combinatorial
Exactly k defective items
Worst-case number of tests

Probabilistic
Each item defective with prob k/n
Typical number of tests

Very sparse
k constant
as $n \to \infty$

Sparse
k grows like n^a
for $a < 1$

Coronavirus in England

England has about 55 million people.

It’s estimated that about 30,000 people currently have COVID-19

Coronavirus in England

England has about 55 million people.

It’s estimated that about 30,000 people currently have COVID-19

Which is the most important calculation?

Coronavirus in England

England has about 55 million people.

It’s estimated that about 30,000 people
currently have COVID-19

Which is the most important calculation?

This is 30,000 infected people,
but the population is irrelevant

Coronavirus in England

England has about 55 million people.

It's estimated that about 30,000 people currently have COVID-19

Which is the most important calculation?

This is 30,000 infected people, but the population is irrelevant

The number of infected people is roughly $(\text{population})^{0.58}$

Coronavirus in England

England has about 55 million people.

It’s estimated that about 30,000 people currently have COVID-19

Which is the most important calculation?

This is 30,000 infected people, but the population is irrelevant

The number of infected people is roughly $(\text{population})^{0.58}$

The number of infected people is roughly 0.05% of the population

Types of group testing

Adaptive
Look at previous tests
before designing the next

Nonadaptive
All tests designed
in advance

Combinatorial
Exactly k defective items
Want to be certain of success

Probabilistic
Each item defective with prob k/n
Average-case number of tests

Very sparse
k constant
as $n \to \infty$

Sparse
k grows like n^a
for $a < 1$

Types of group testing

Adaptive
Look at previous tests
before designing the next

Nonadaptive
All tests designed
in advance

Combinatorial
Exactly k defective items
Want to be certain of success

Probabilistic
Each item defective with prob k/n
Average-case number of tests

Very sparse
k constant
as $n \to \infty$

Sparse
k grows like n^a
for $a < 1$

Linear
$k \sim pn$
grows linearly with n

Mathematicians like to think that "sparse" means $\boldsymbol{k} = \boldsymbol{o}(\boldsymbol{n})$.

But consider if $\boldsymbol{k}$ being "small but linear in $\boldsymbol{n}$" might be more relevant in the real world.

2

Lower bounds

Lower bound

For successful group testing, we need

$$T \geq \log_2 \binom{n}{k} \text{ tests}$$

Lower bound

For successful group testing, we need

$$T \geq \log_2 \binom{n}{k} \text{ tests}$$

Proof for combinatorialists:

There are $\binom{n}{k}$ possible defective sets.

There are up to 2^T sequences of test results.

Each possible defective set needs
a unique outcome sequence of test results.

Lower bound

For successful group testing, we need
$T \geq \log_2 \binom{n}{k}$ tests

Proof for information theorists:

We need $\log_2 \binom{n}{k}$ bits of information
to define the defective set.

We can get at most 1 bit of information
from each test.

Lower bound

$$T \geq \log_2 \binom{n}{k}$$

Very sparse regime (k constant):

$$\log_2 \binom{n}{k} \sim k \log_2 n$$

Lower bound

$$T \geq \log_2 \binom{n}{k}$$

Very sparse regime (k constant):

$$\log_2 \binom{n}{k} \sim k \log_2 n$$

Sparse regime ($k = n^a$):

$$\log_2 \binom{n}{k} \sim k \log_2 \frac{n}{k} = (1-a)k \log_2 n$$

Lower bound

$$T \geq \log_2 \binom{n}{k}$$

Very sparse regime (k constant):

$$\log_2 \binom{n}{k} \sim k \log_2 n$$

Sparse regime (k = n^a):

$$\log_2 \binom{n}{k} \sim k \log_2 \frac{n}{k} = (1-a) k \log_2 n$$

Linear regime (k = pn):

$\log_2 \binom{n}{k} \sim H(p)n$ where $H(p)$ is the binary entropy

Individual testing

$$T = n$$

Very sparse regime (k constant):

$$\log_2 \binom{n}{k} \sim k \log_2 n$$

Sparse regime ($k = n^a$):

$$\log_2 \binom{n}{k} \sim k \log_2 \frac{n}{k} = (1-a)k \log_2 n$$

Linear regime ($k = pn$):

$\log_2 \binom{n}{k} \sim H(p)n$ where $H(p)$ is the binary entropy

In the linear regime,
naïve “sparsity-ignorant” algorithms
can be competitive
or even optimal.

.

In the linear regime,
naïve “sparsity-ignorant” algorithms
can be competitive
or even optimal.

In the linear regime,
order-optimal behaviour
can be obvious;
try to find the constants.

3

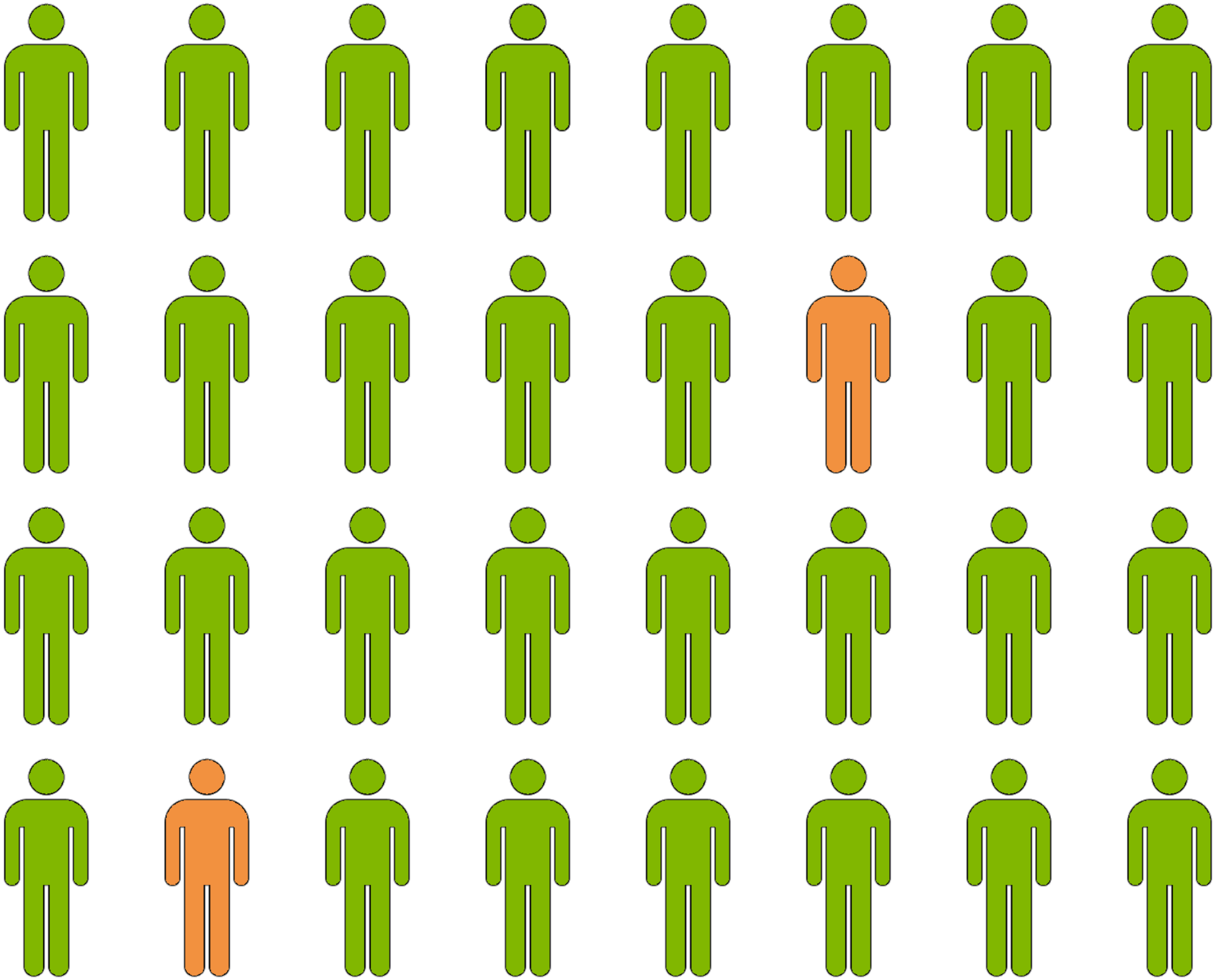
Algorithms for adaptive group testing

Binary splitting

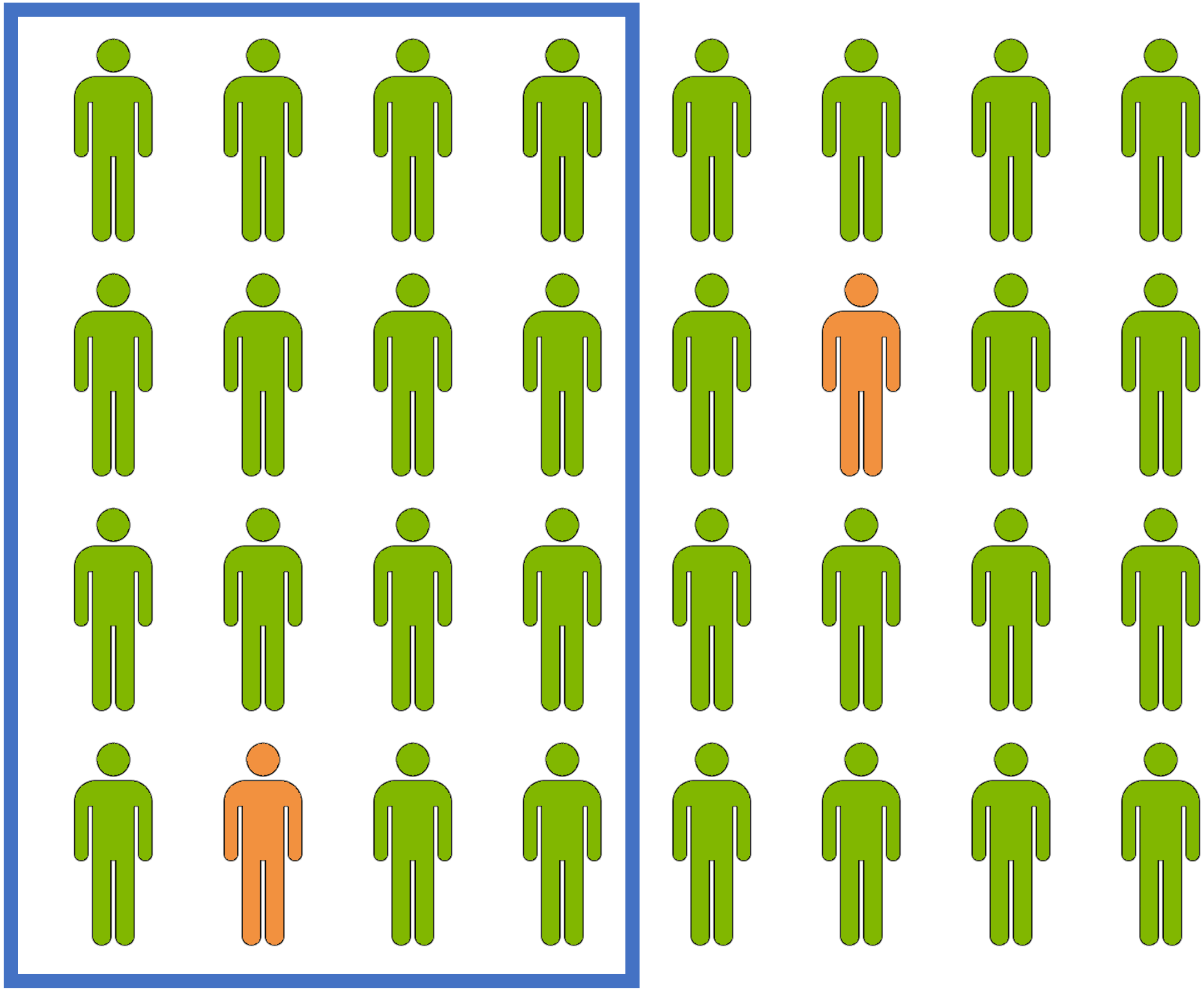
(Sobel & Groll, 1959)

Keep splitting the set in half,
keeping a half that has
a defective item in it

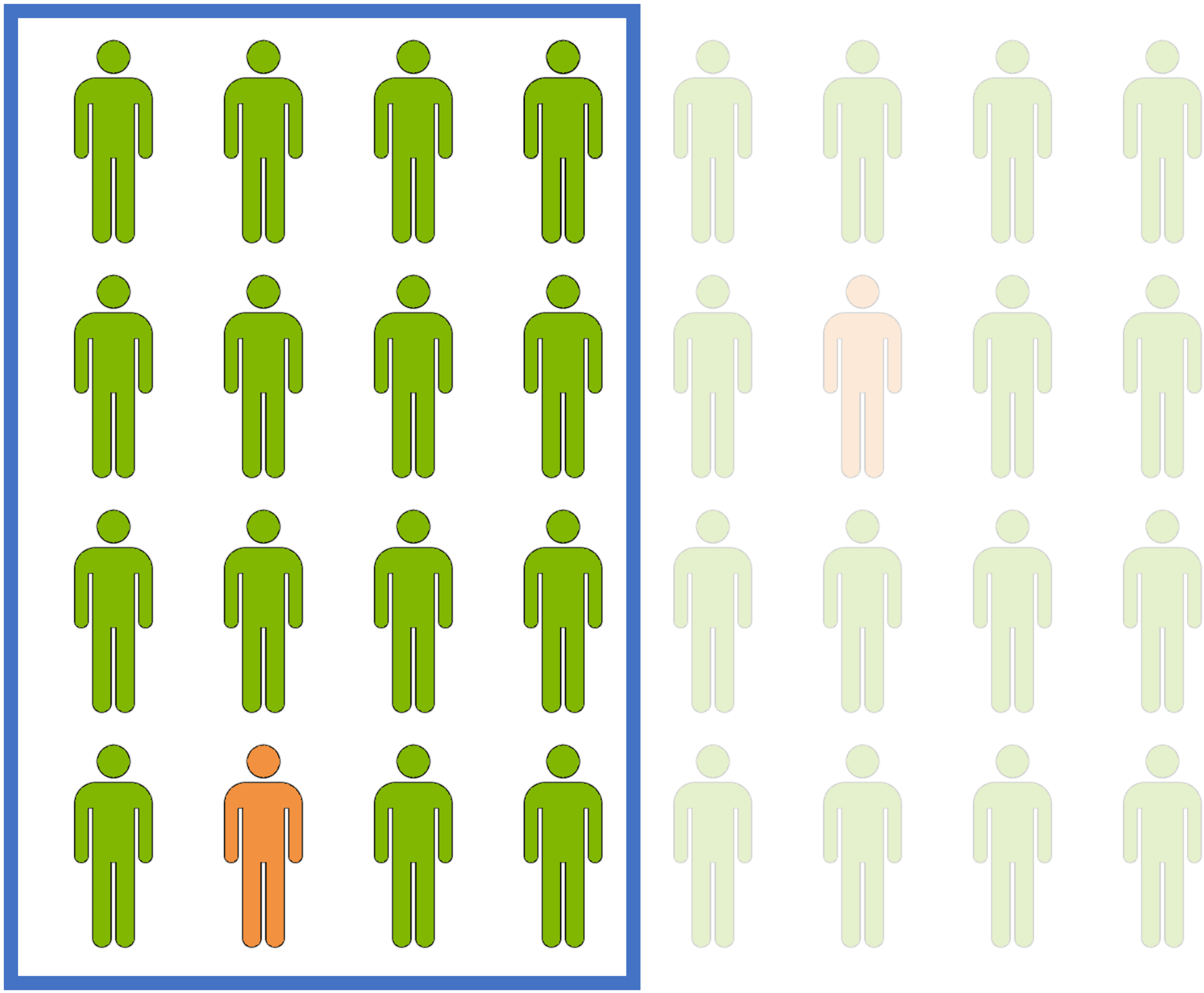
Binary splitting



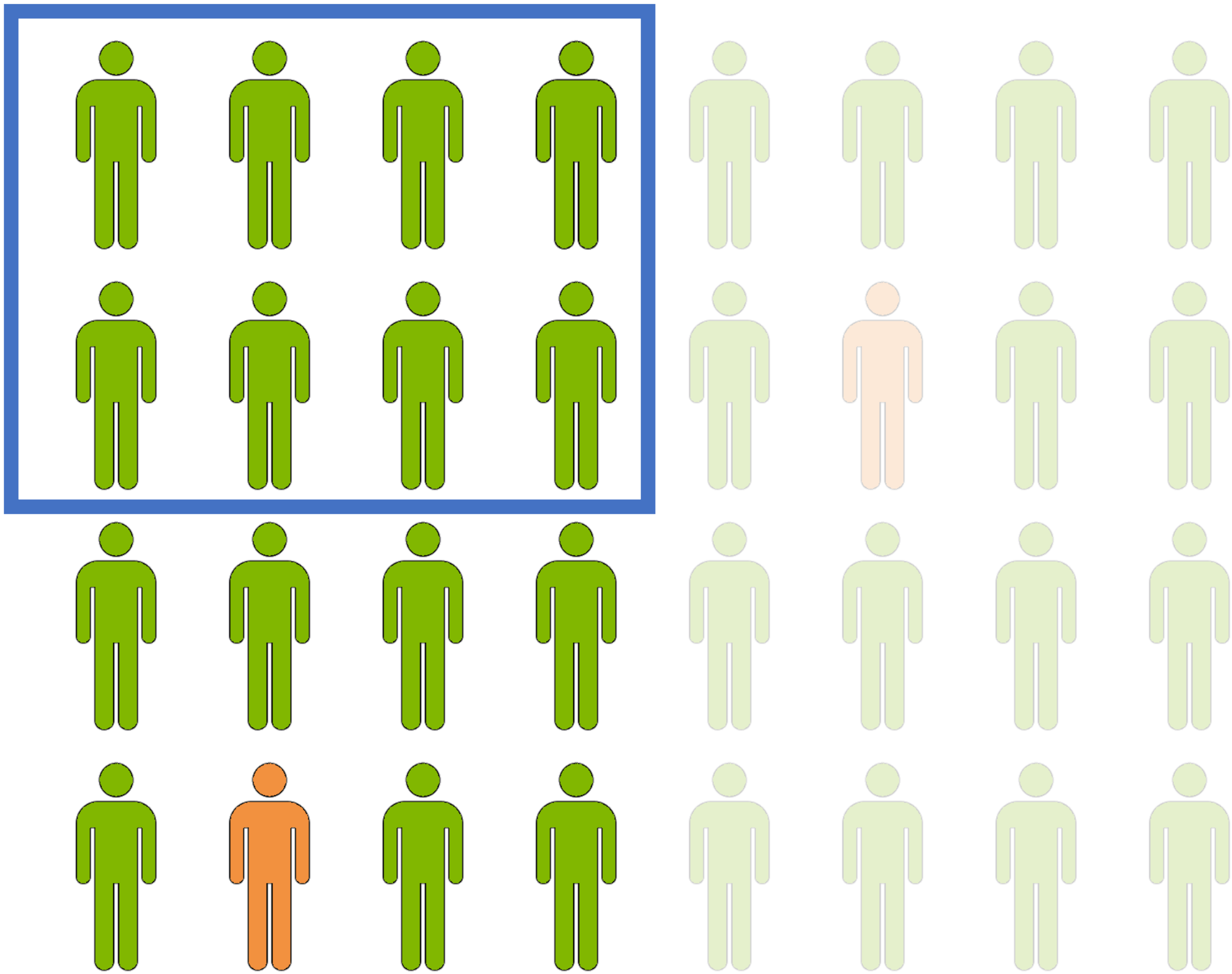
Binary splitting



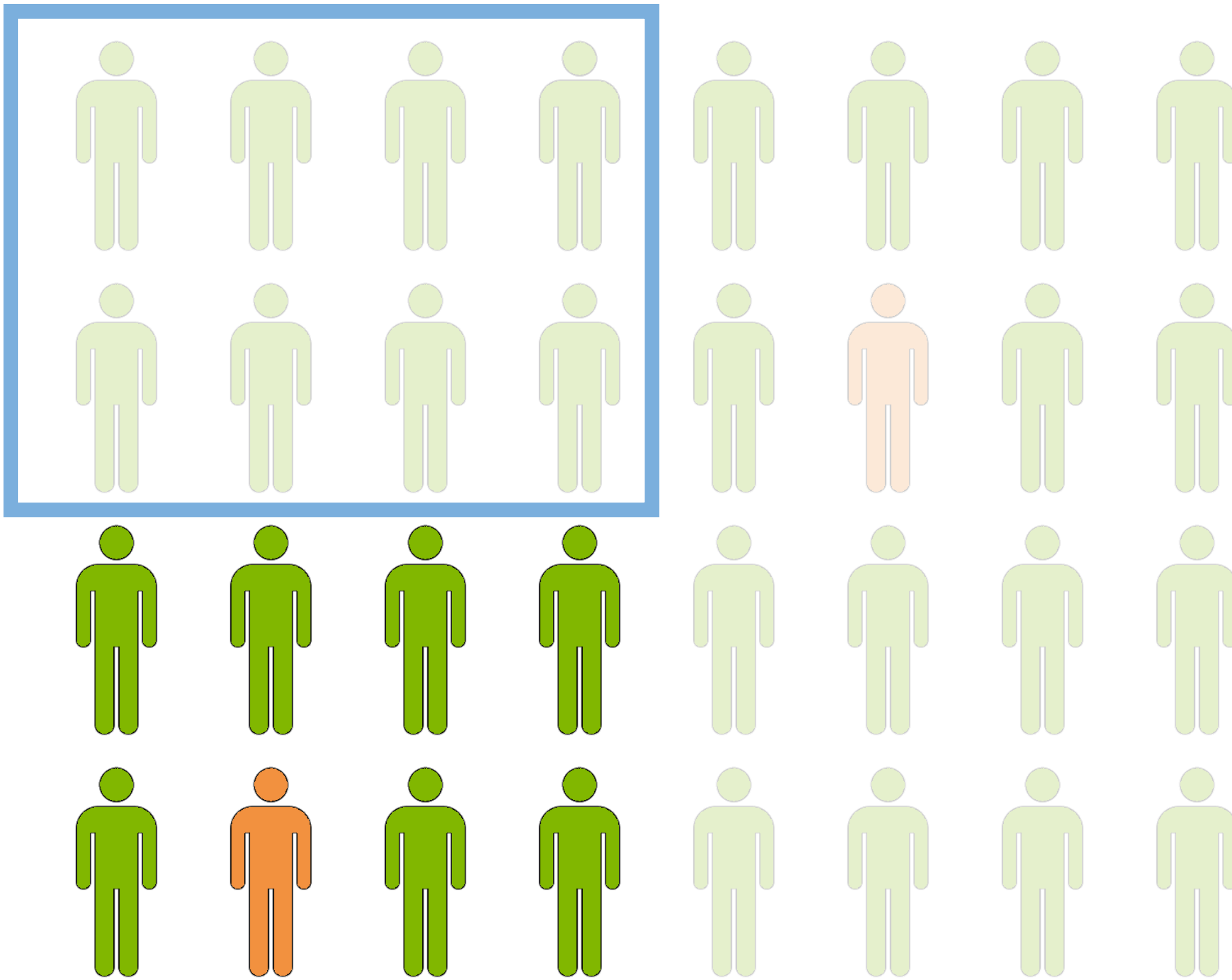
Binary splitting



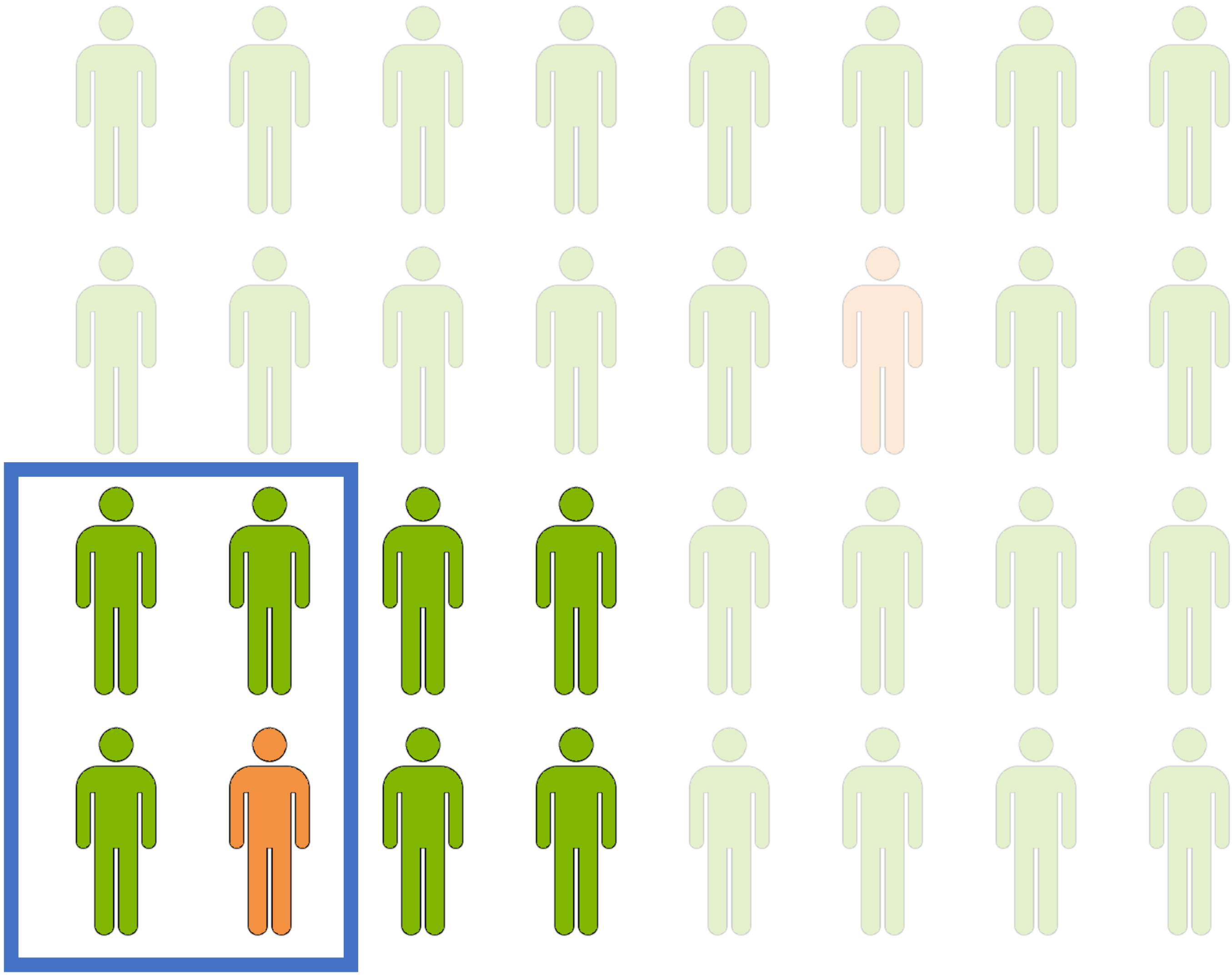
Binary splitting



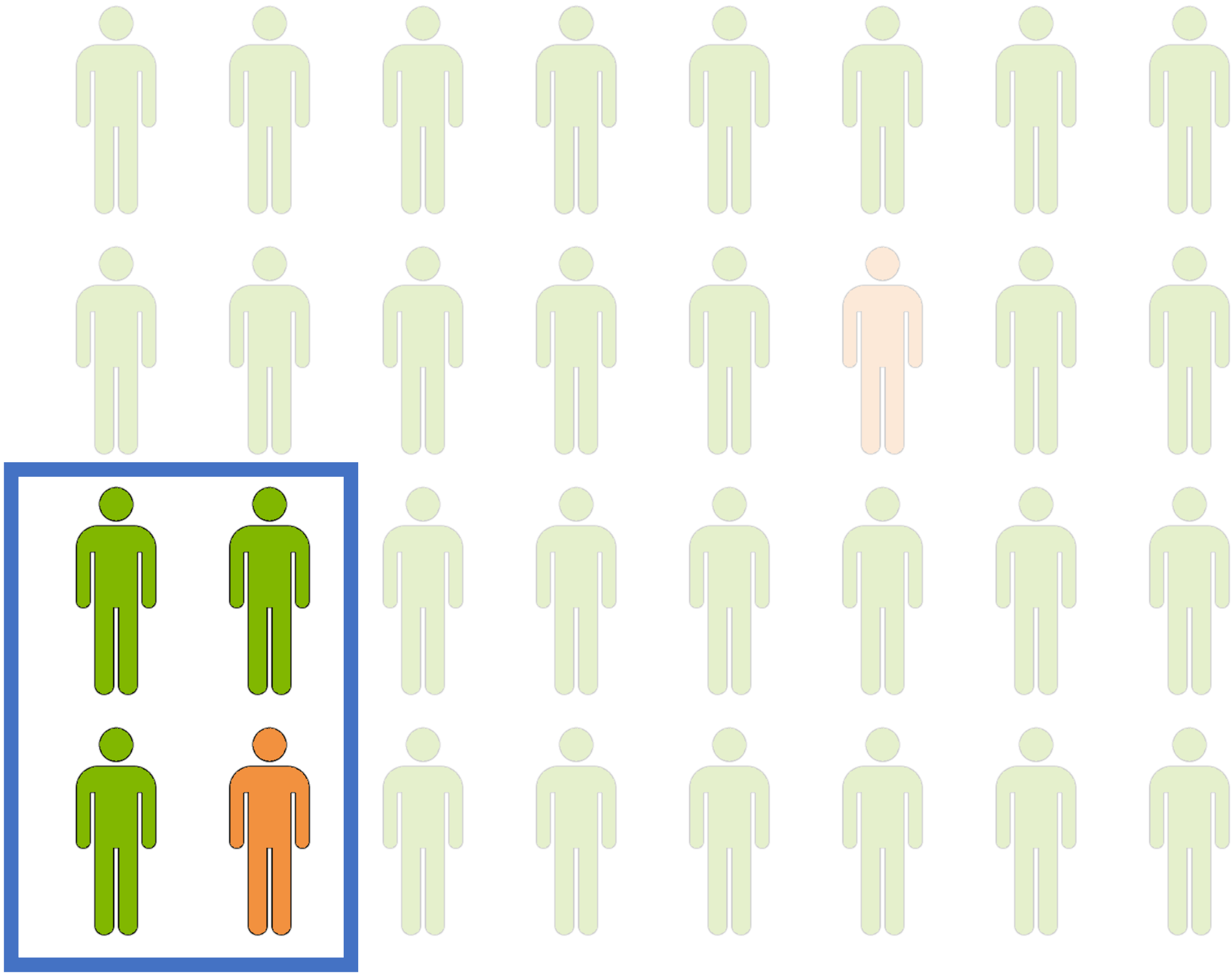
Binary splitting



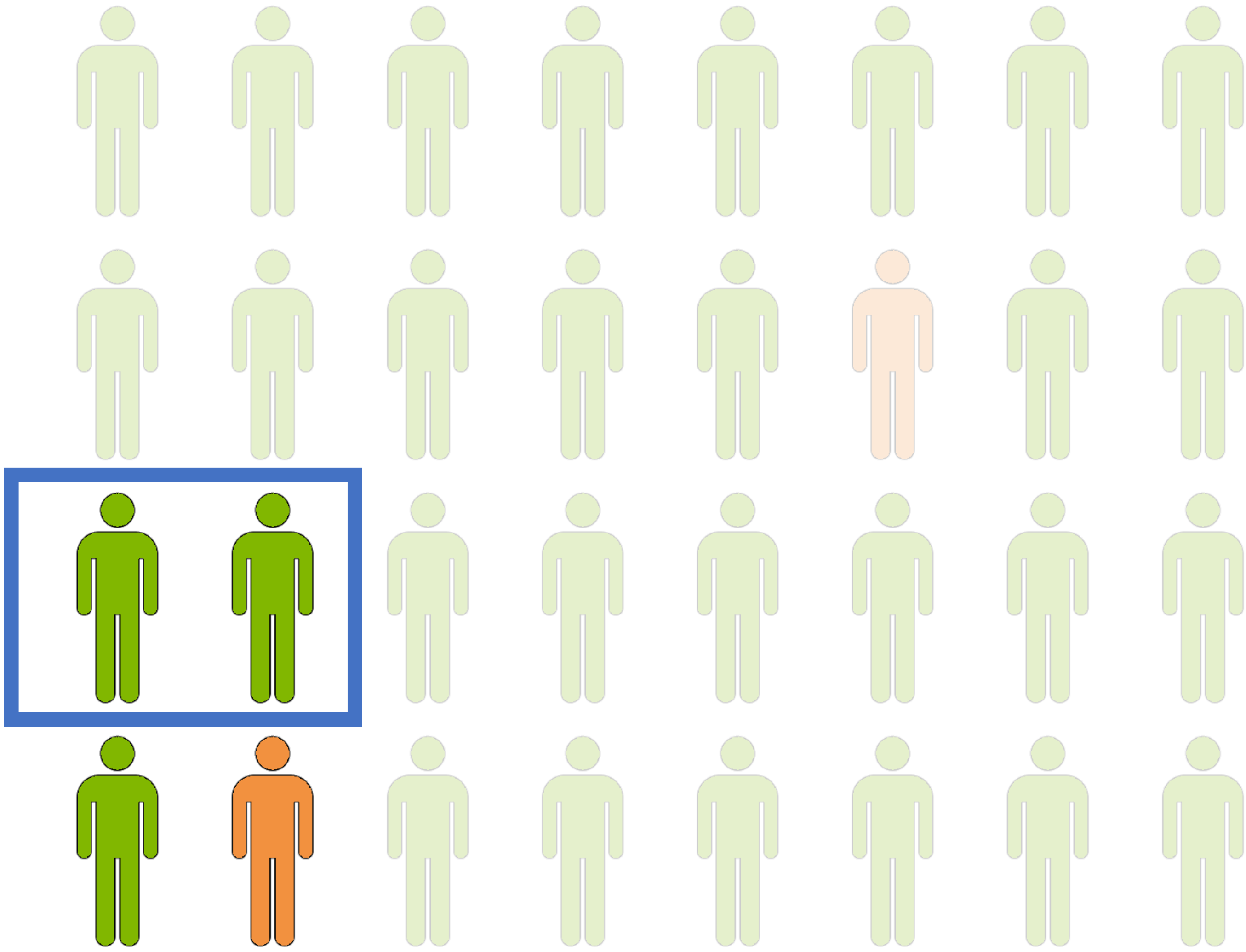
Binary splitting



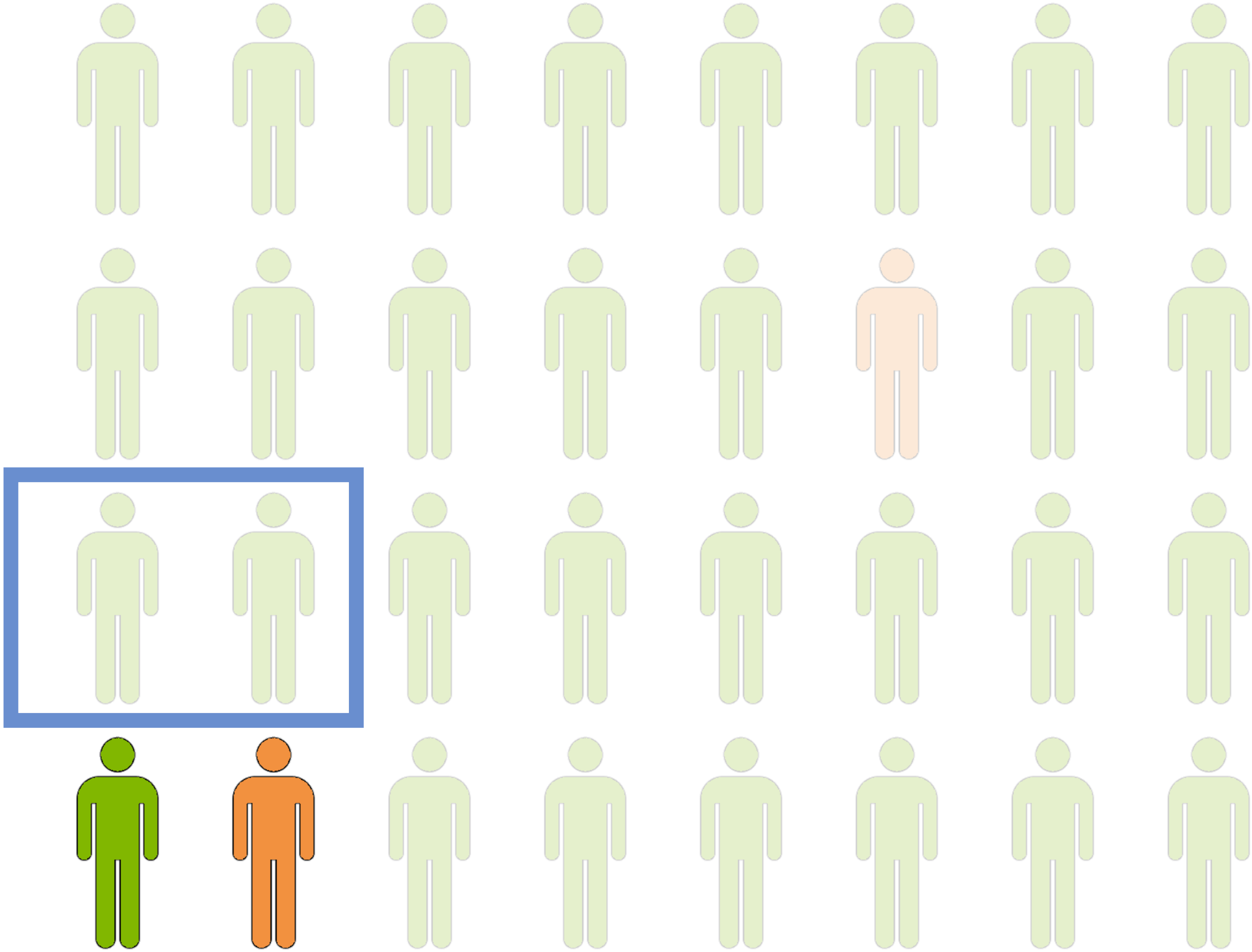
Binary splitting



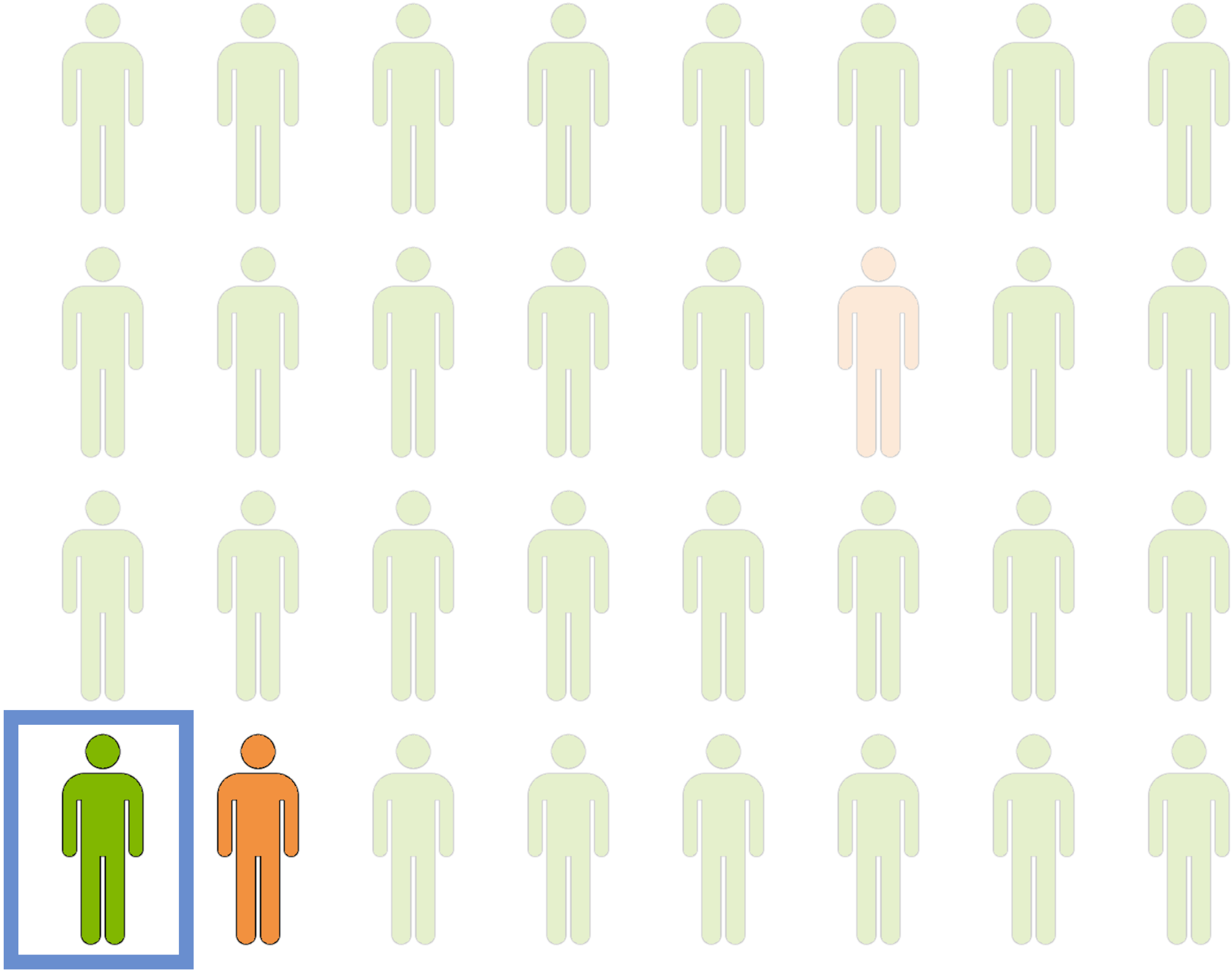
Binary splitting



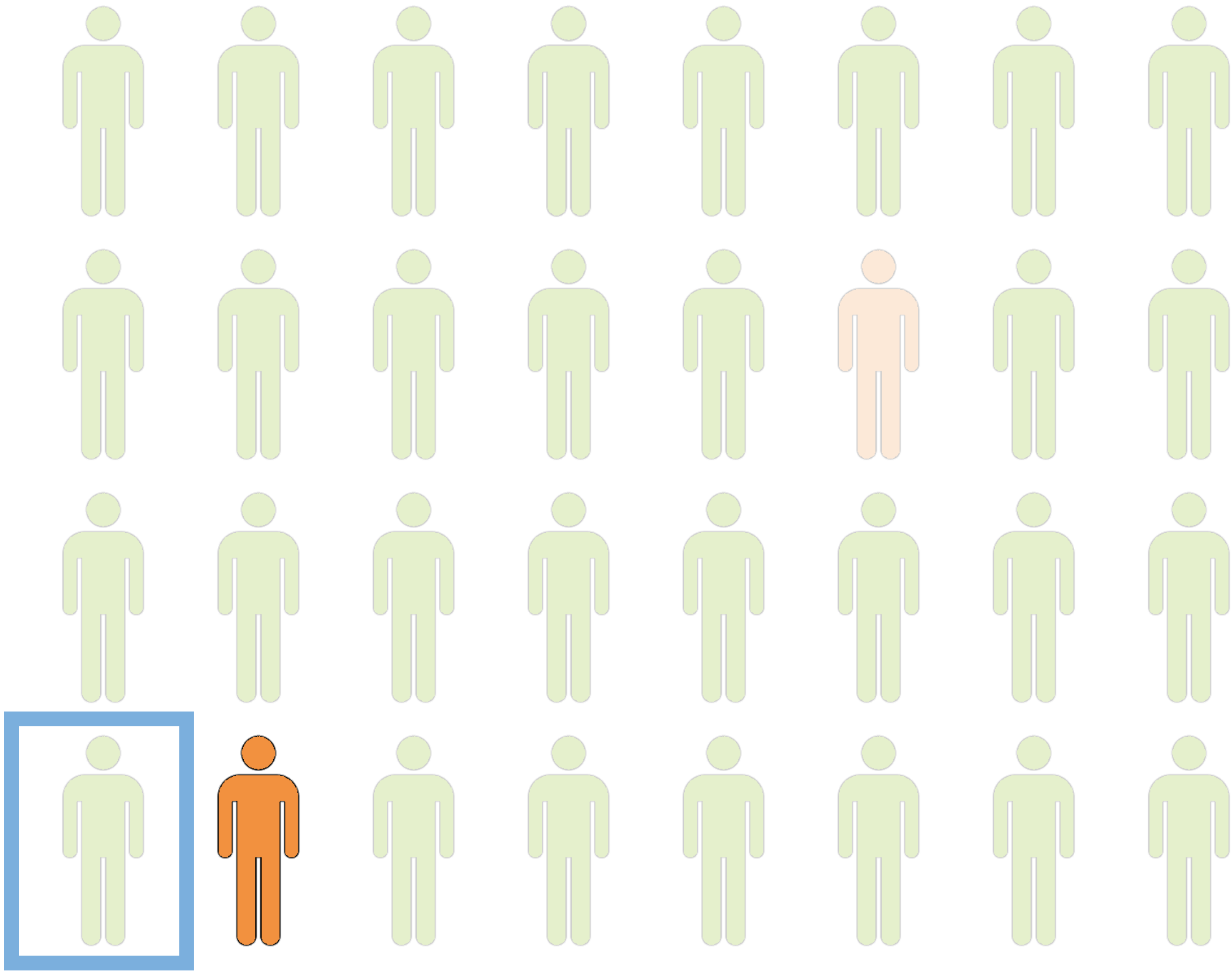
Binary splitting



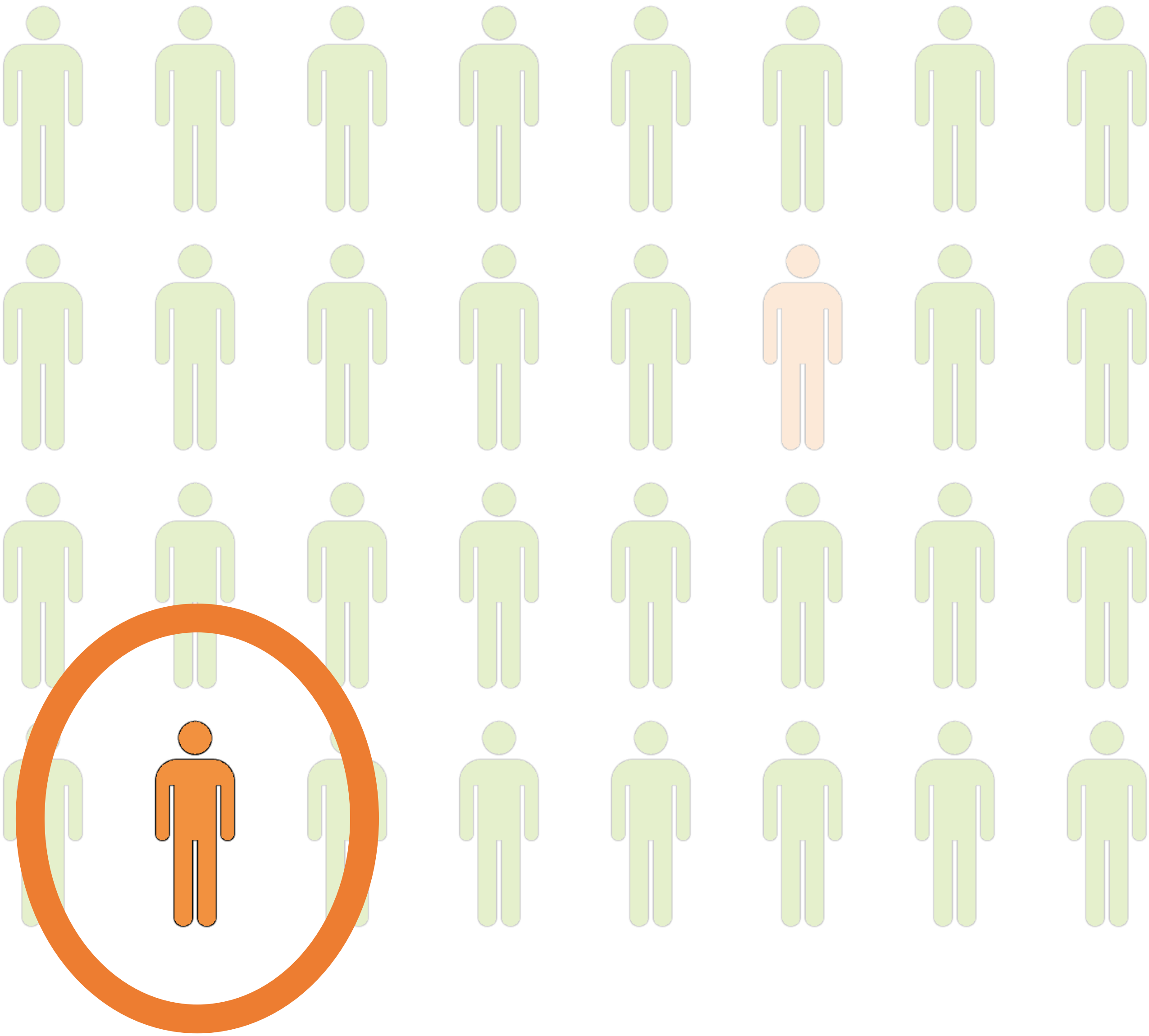
Binary splitting



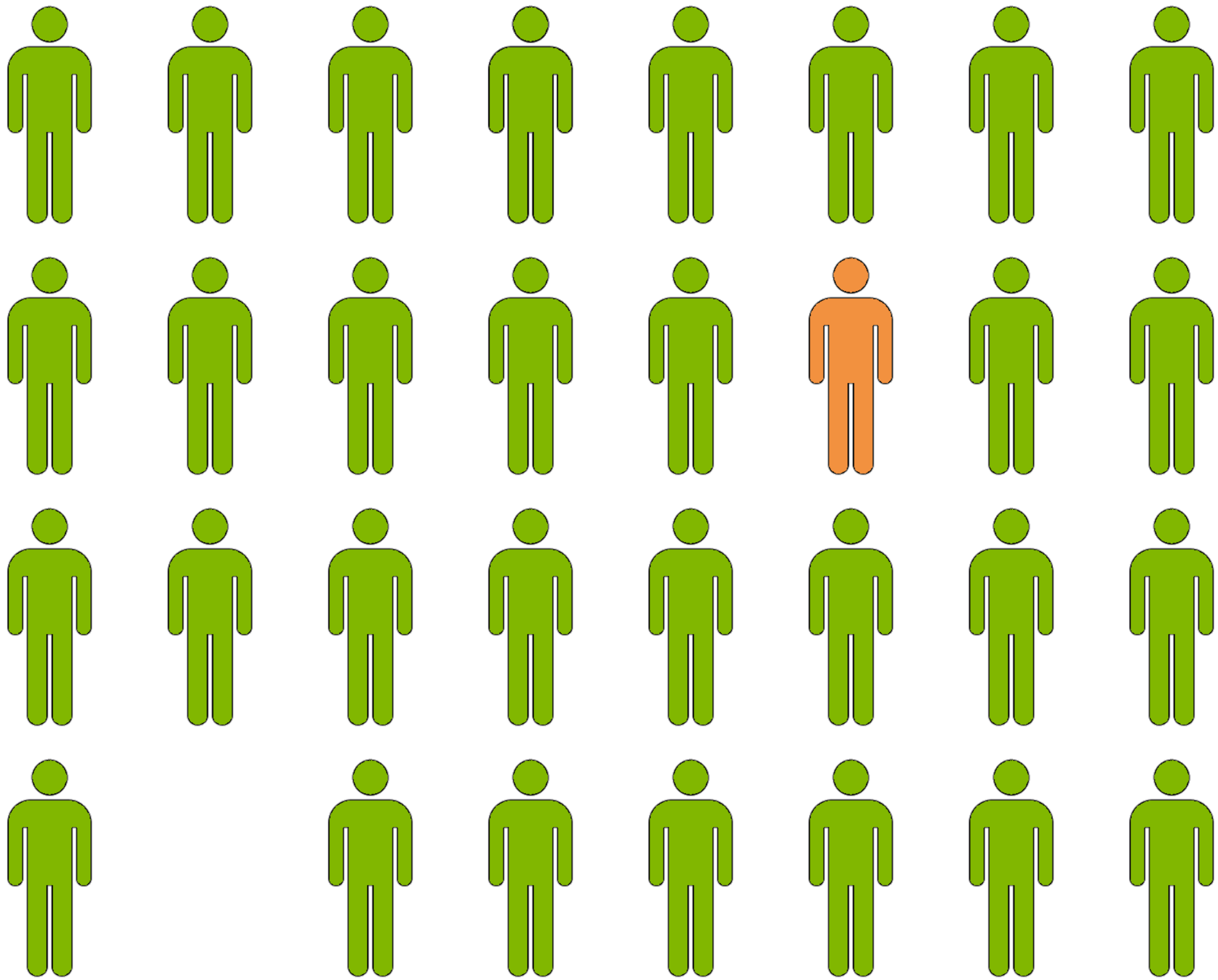
Binary splitting



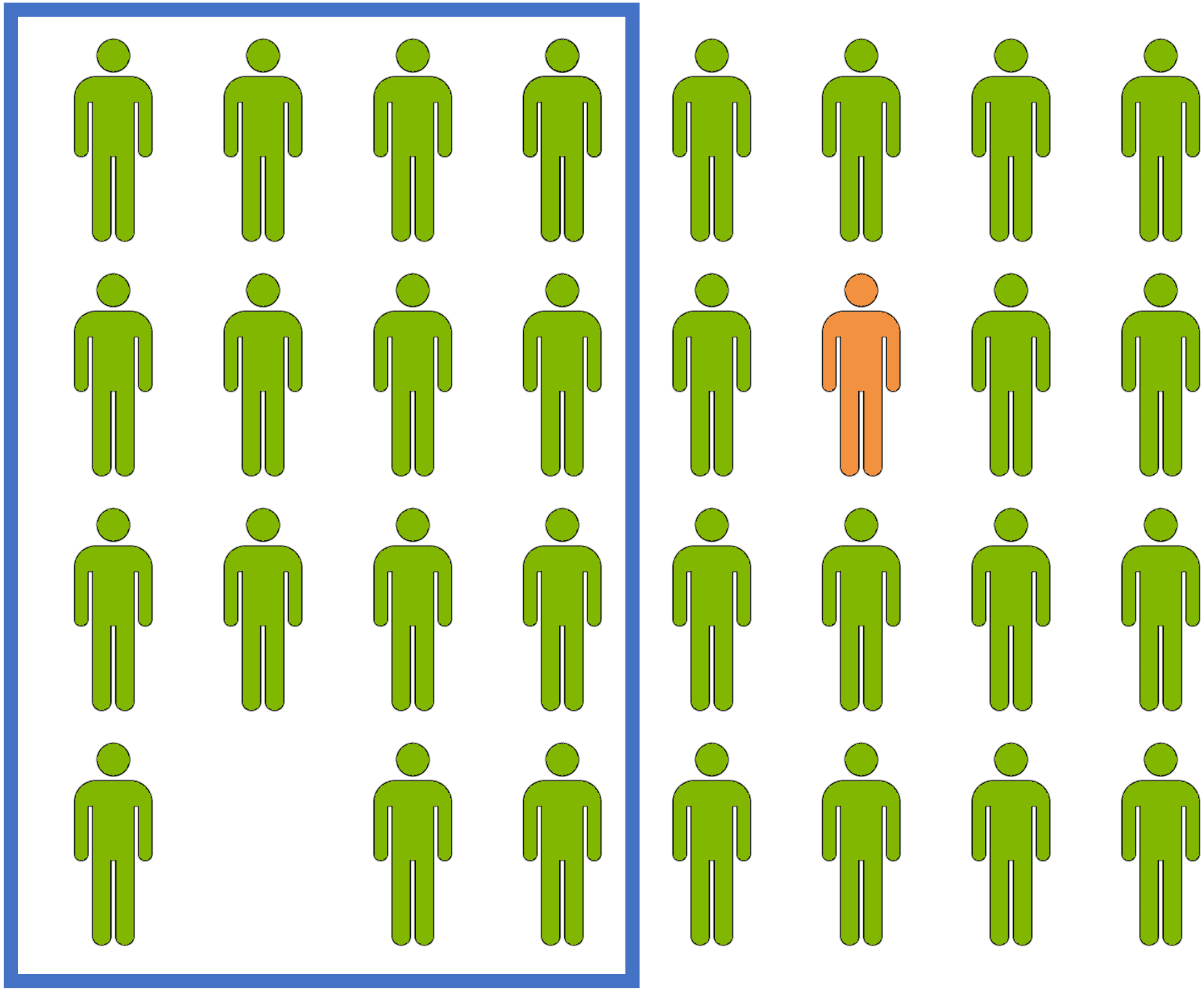
Binary splitting



Binary splitting



Binary splitting



Simple binary splitting

(Sobel & Groll, 1959)

For the combinatorial (*k* known) model:

Repeat k times:

Use **binary splitting** to find a defective.
Remove it.

Simple binary splitting

(Sobel & Groll, 1959)

For the probabilistic (*k* unknown) model:

1) Test the whole set.

If the test is positive:
Use **binary splitting** to find a defective.
Remove it, and return to 1).

If the test is negative:
All items are nondefective. Halt.

Simple binary splitting

Theorem:

The simple binary splitting algorithm requires

$$k \log_2 n + O(k)$$

tests.

Simple binary splitting

Theorem:

The simple binary splitting algorithm requires

$$k \log_2 n + O(k)$$

tests.

k rounds of binary splitting
a set of size n

"book-keeping" tests
and rounding errors

Simple binary splitting

Theorem:

The simple binary splitting algorithm requires

$$k \log_2 n + O(k)$$

tests.

Very sparse regime: optimal scaling and constant

Sparse regime: optimal scaling; suboptimal constant

Linear regime: worse than individual testing for large n

Generalized binary splitting

(Hwang, 1972)

In the sparse and linear regimes,
we waste too much time
at the beginning of each stage
testing sets that are
almost certain to contain a defective item

Generalized binary splitting

(Hwang, 1972)

Split into k sets of size n/k

For each set do simple binary splitting:

1) Test the whole set.

If the test is positive:
Use **binary splitting** to find a defective.
Remove it, and return to 1).

If the test is negative:
All items are nondefective. Halt.

Generalized binary splitting

(Hwang, 1972)

On average, one defective each

Split into k sets of size n/k

For each set do simple binary splitting:

1) Test the whole set.

If the test is positive:
Use **binary splitting** to find a defective.
Remove it, and return to 1).

If the test is negative:
All items are nondefective. Halt.

Generalized binary splitting

(Hwang, 1972; Baldassini–Johnson–Aldridge, 2013)

Theorem:

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k)$$

tests.

This is optimal in the sparse regime.

Generalized binary splitting

(Hwang, 1972; Baldassini–Johnson–Aldridge, 2013)

Theorem:

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k)$$

tests.

This is optimal in the sparse regime.

k **rounds of binary splitting a set of size** n/k

"book-keeping" tests and **rounding errors**

Don’t waste effort
on measurements
if you think you know
what the answer will be.

Linear regime

Split into k sets of size n/k

For each set do simple binary splitting.

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k)$$

tests

Linear regime

Split into k sets of size $n/k = 1/p$

For each set do simple binary splitting.

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k) = \left(p \log_2 \frac{1}{p}\right) n + O(n)$$

tests

Linear regime

Might not be
an integer

Split into k sets of size $n/k = 1/p$

For each set do simple binary splitting.

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k) = \left(p \log_2 \frac{1}{p}\right) n + O(n)$$

tests

Linear regime

Might not be an integer

Split into k sets of size $n/k = 1/p$

For each set do simple binary splitting.

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k) = \left(p \log_2 \frac{1}{p}\right) n + O(n)$$

tests

Need to be careful with “error term”

Linear regime

Might not be an integer

Split into k sets of size $n/k = 1/p$

For each set do simple binary splitting.

The generalized binary splitting algorithm requires

$$k \log_2 \frac{n}{k} + O(k) = \left(p \log_2 \frac{1}{p}\right) n + O(n)$$

tests

Suboptimal compared to counting bound

Need to be careful with “error term”

Instead we'll try…

(Parameter to be chosen later)

1) Pick a set of size $m = 2^s$.

2) Test the set.

If the test is positive:
Use **binary splitting** to find a defective.
Return to 1).

If the test is negative:
All items in the set are nondefective.
Return to 1).

Instead we'll try…

1) Pick a set of size $m = 2^s$.

$m = 1$: Individual testing

2) Test the set. $m = 2$: Fischer–Klasner–Wegenera, 1999

If the test is positive:
Use **binary splitting** to find a defective.
Return to 1).

If the test is negative:
All items in the set are nondefective.
Return to 1).

Combinatorial testing

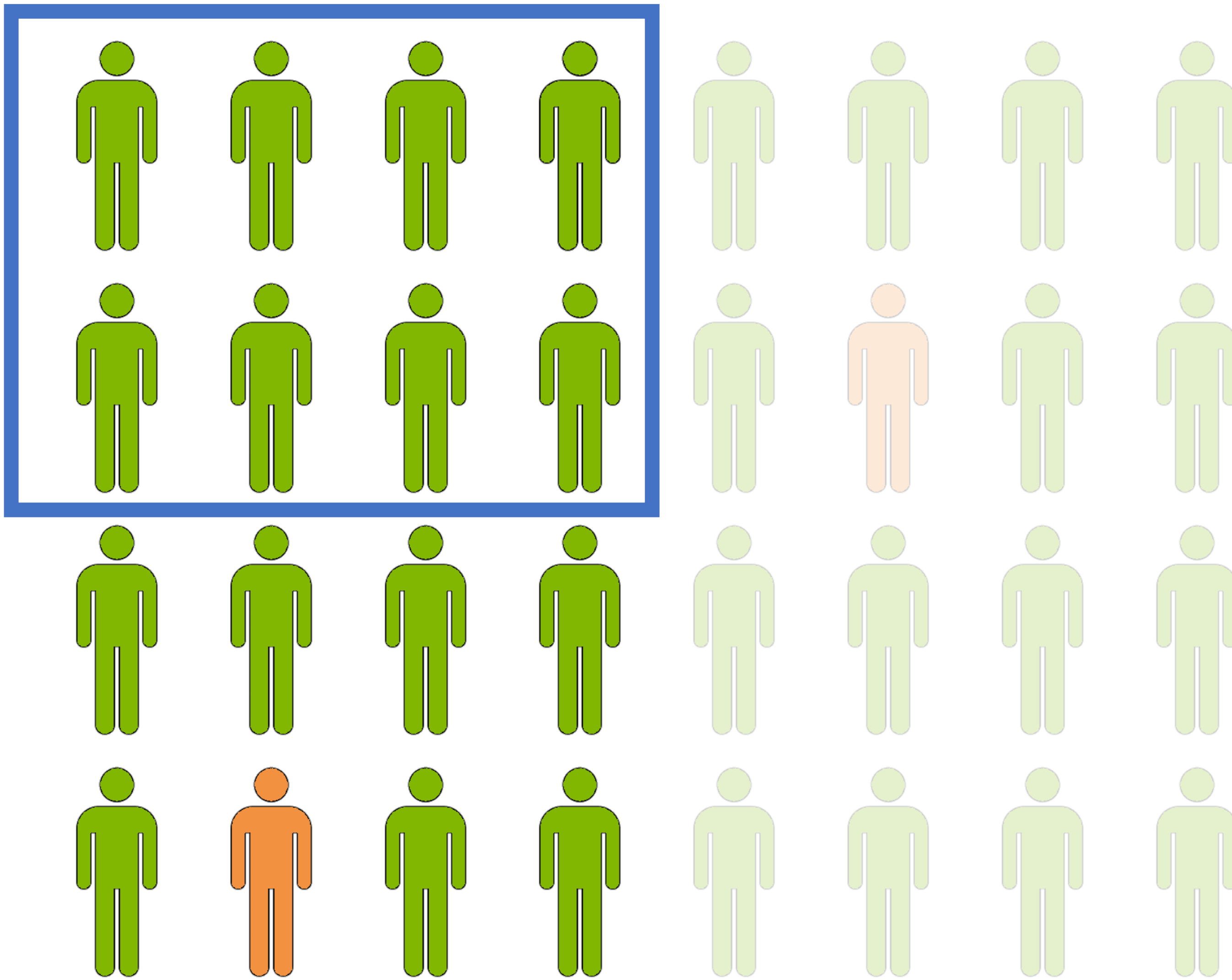
At each run through the loop we find:

$m = 2^s$ nondefectives
in 1 test

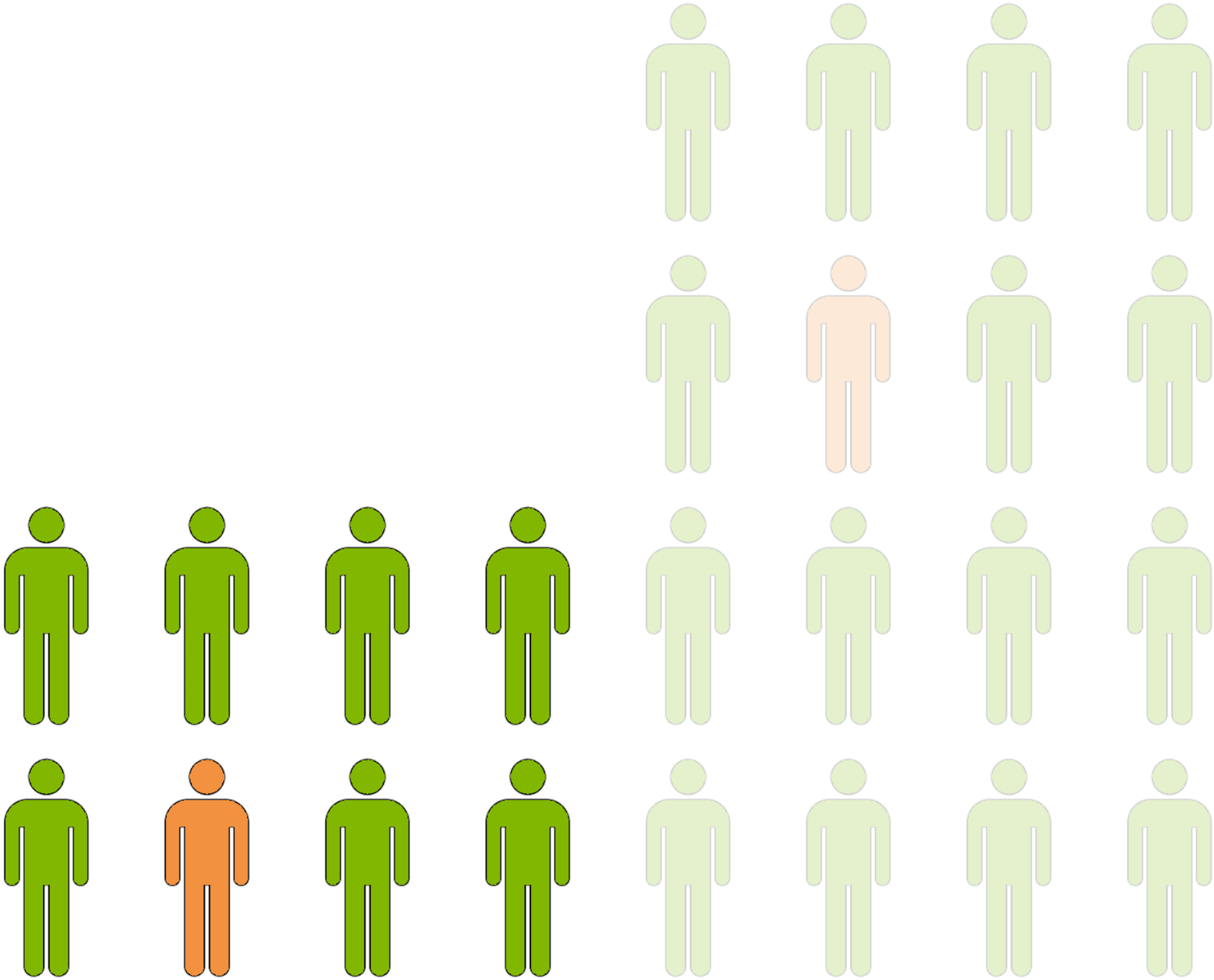
or

1 defective
in $1 + \log_2 m = s + 1$ tests

Binary splitting



Binary splitting



Combinatorial testing

At each run through the loop we find:

$m = 2^s$ nondefectives
in 1 test

or

1 defective

in $1 + \log_2 m = s + 1$ tests

Combinatorial testing

At each run through the loop we find:

$m = 2^s$ nondefectives
in 1 test

or

1 defective
and up to $m - 1 = 2^s - 1$ nondefectives
in $1 + \log_2 m = s + 1$ tests

Combinatorial testing

At each run through the loop we find:

$m = 2^s$ nondefectives
in 1 test

or

Worst-case analysis:
Assume we're unlucky

1 defective
~~and up to $m - 1 = 2^s - 1$ nondefectives~~
in $1 + \log_2 m = s + 1$ tests

Combinatorial testing

Each of the k defectives requires
$1 + \log_2 m = s + 1$ tests.

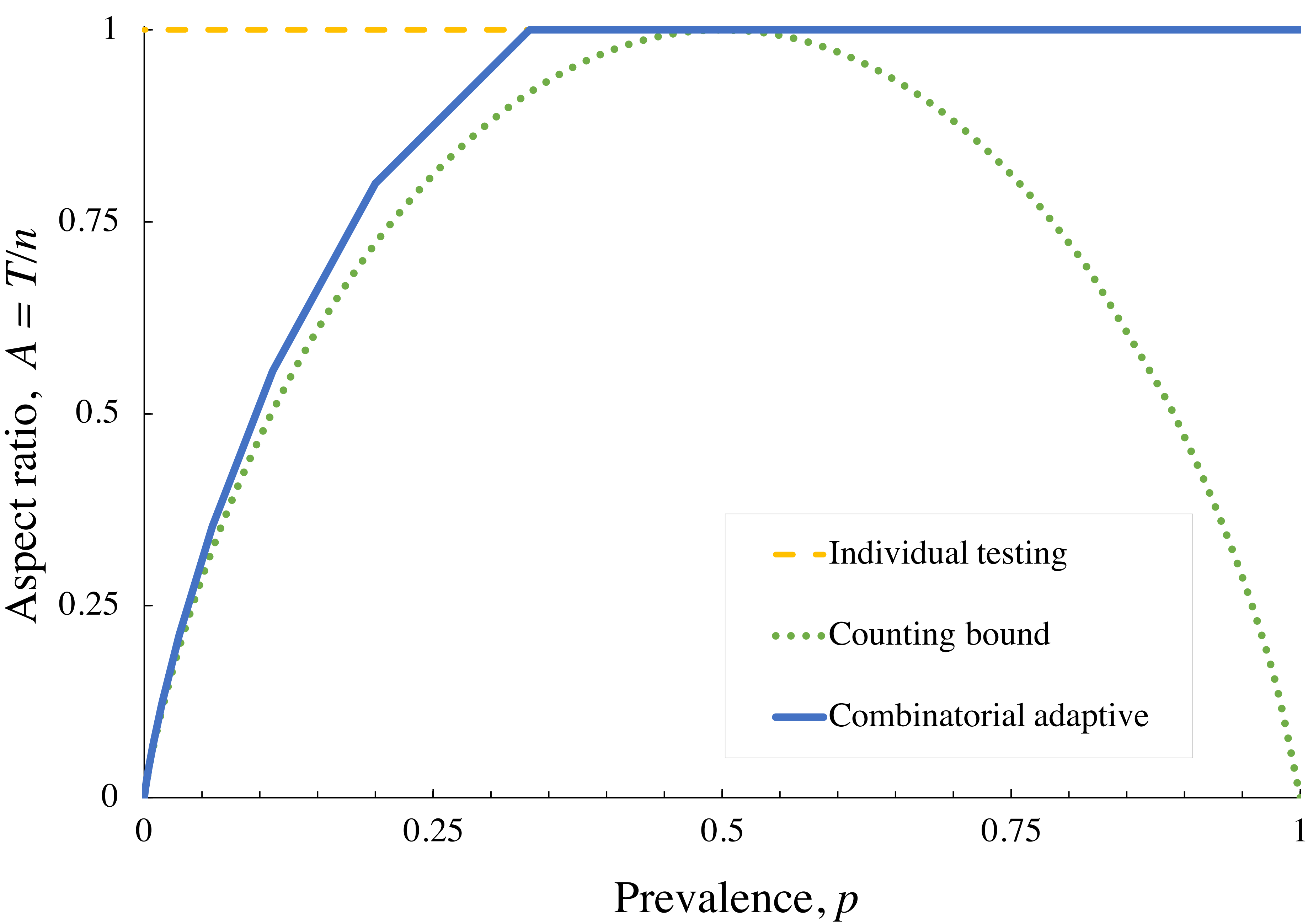
Each set of $m = 2^s$ nondefectives
requires 1 test.

Combinatorial testing

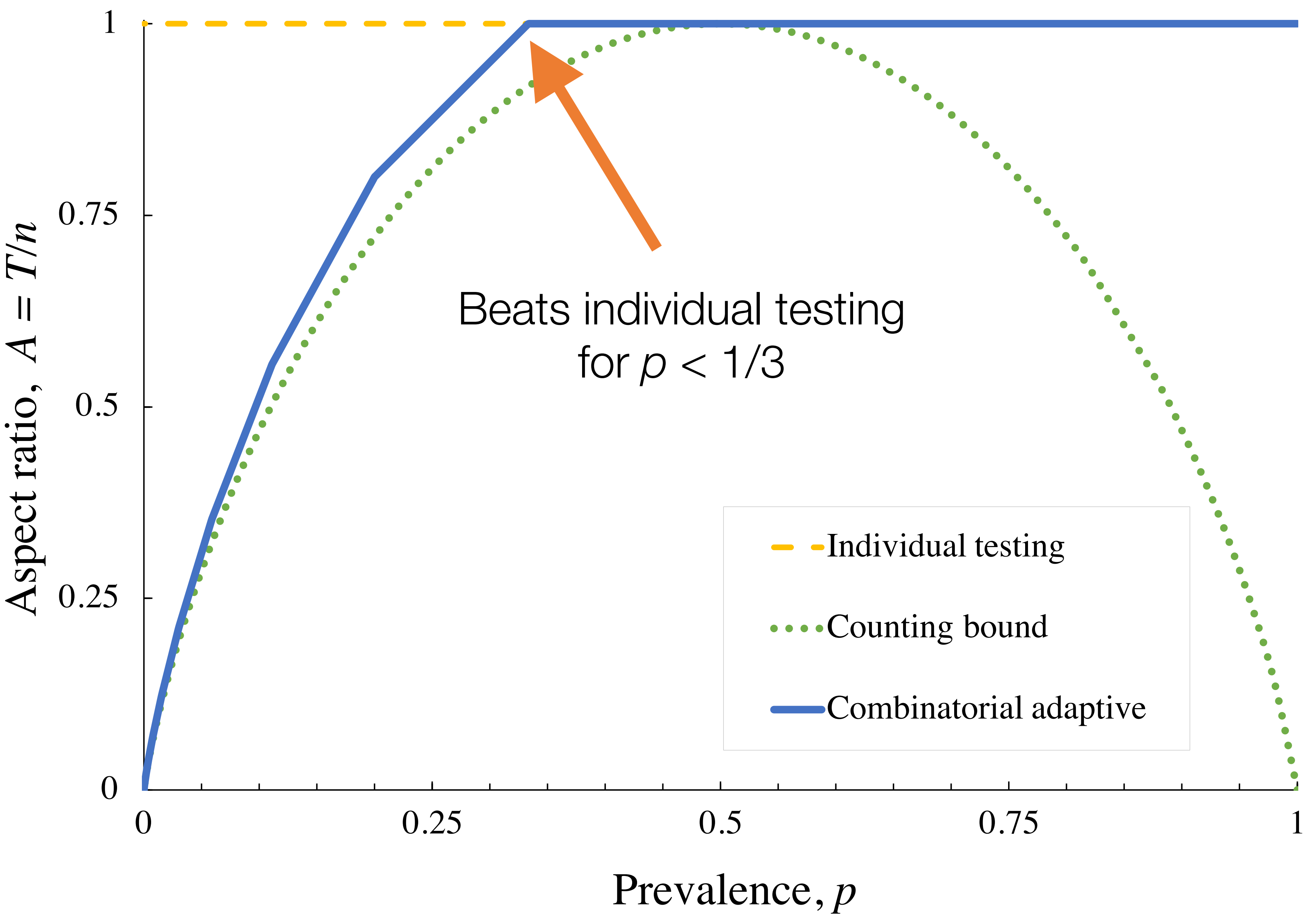
Each of the k defectives requires $1 + \log_2 m = s + 1$ tests.

Each set of $m = 2^s$ nondefectives requires 1 test.

$$
\begin{aligned}
T &= (s+1)k + \frac{1}{2^s}(n-k) \\
&= \left((s+1)p + \frac{1}{2^s}(1-p)\right)n
\end{aligned}
$$



Aspect ratio, $A = T/n$
1
0.75
0.5
0.25
0
0
0.25
0.5
0.75
1
Prevalence, p
Individual testing
Counting bound
Combinatorial adaptive



1
0.75
0.5
0.25
0
Aspect ratio, $A = T/n$
0
0.25
0.5
0.75
1
Prevalence, p
Beats individual testing for $p < 1/3$
Individual testing
Counting bound
Combinatorial adaptive

Open problem

Prove that individual testing
is optimal for $p \geq 1/3$
for combinatorial testing.

(Conjectured by Hu–Hwang–Wang, 1981)

Probabilistic testing

At each run through the loop we find:

$m = 2^s$ nondefectives
in 1 test

or

1 defective
and up to $m - 1 = 2^s - 1$ nondefectives
in $1 + \log_2 m = s + 1$ tests

Probabilistic testing

At each run through the loop we find:

$m = 2^a$ nondefectives

in 1 test

or

How well do we do on average?

1 defective

and up to $m - 1 = 2^a - 1$ nondefectives

in $1 + \log_2 m = a + 1$ tests

The algorithm

Aldridge (2019) shows that for $q = 1 - p$:

Average tests per loop:

$$F = q^m \times 1 + (1 - q^m)(1 + \log_2 m)$$

The algorithm

Aldridge (2019) shows that for $q = 1 - p$:

Average tests per loop:

$$F = q^m \times 1 + (1 - q^m)(1 + \log_2 m)$$

Average number of items classified per loop:

$$G = mq^m + \sum_{j=1}^{m} jpq^j$$

The algorithm

Aldridge (2019) shows that for $q = 1 - p$:

Average tests per loop:

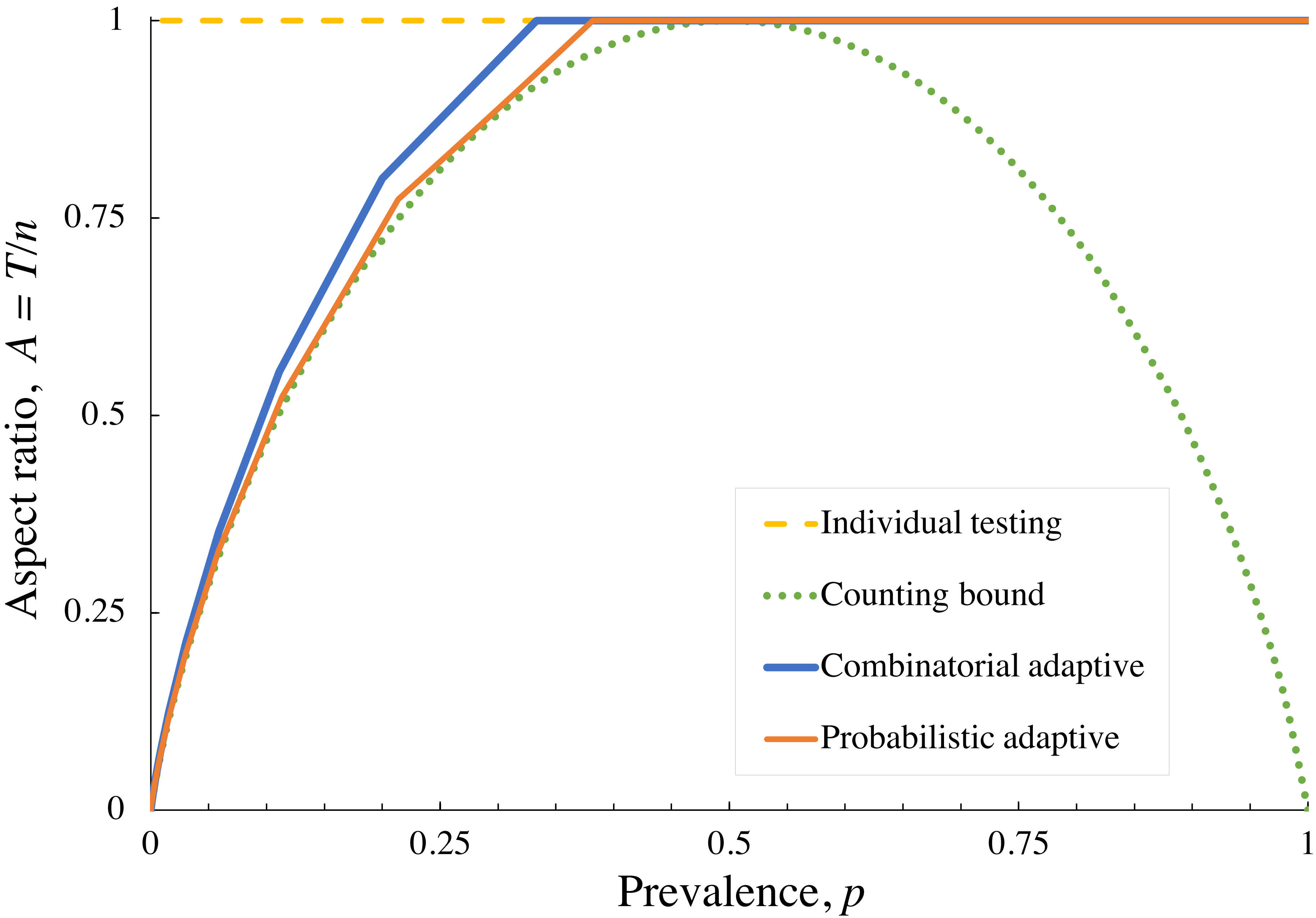
$$F = q^m \times 1 + (1 - q^m)(1 + \log_2 m)$$

Average number of items classified per loop:

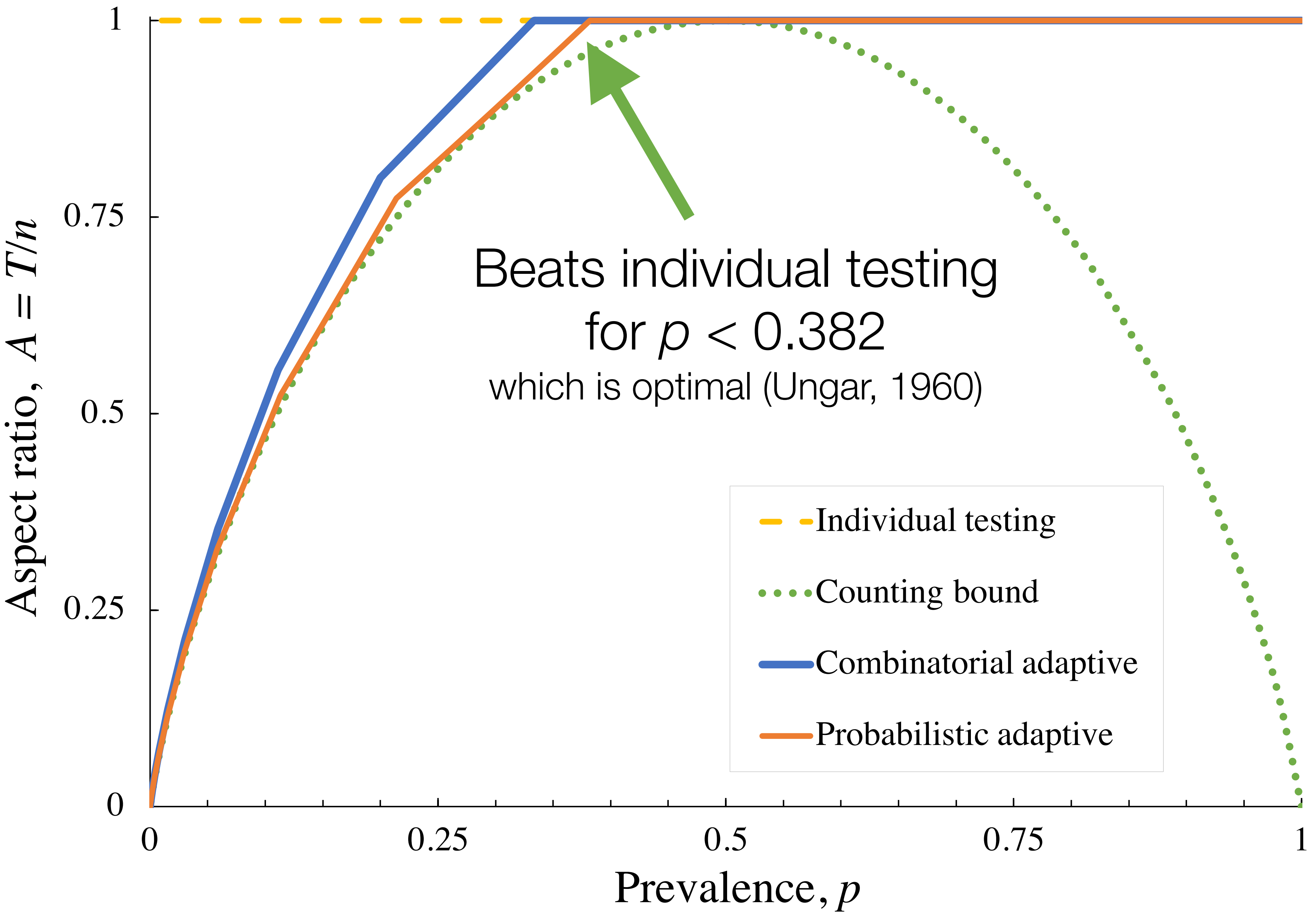
$$G = mq^m + \sum_{j=1}^{m} jpq^j$$

Average number of tests to classify all items:

$$T = Fn/G$$



1
0.75
0.5
0.25
0
Aspect ratio, $A = T/n$
0
0.25
0.5
0.75
1
Prevalence, p
Individual testing
Counting bound
Combinatorial adaptive
Probabilistic adaptive

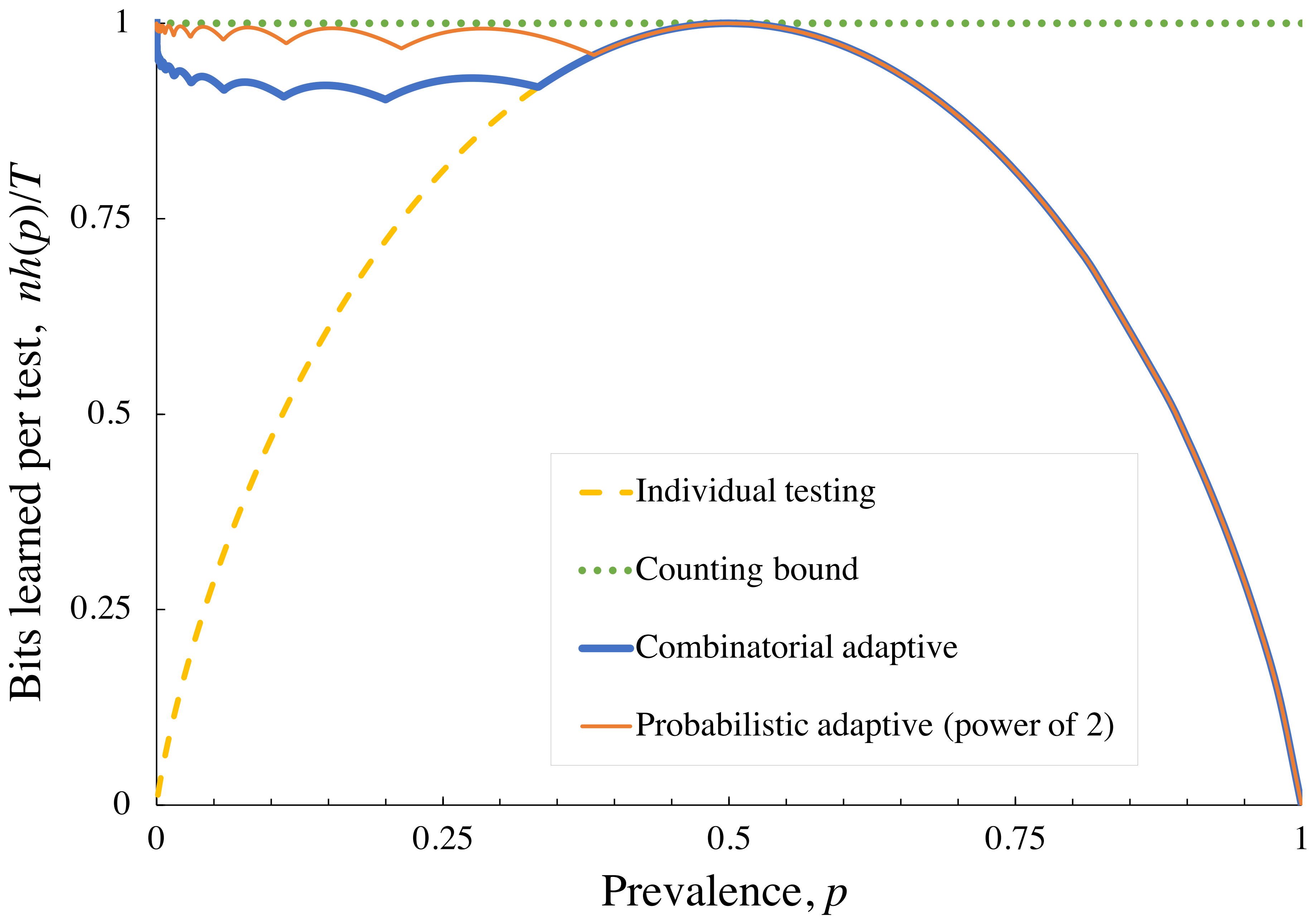

Beats individual testing
for p < 0.382
which is optimal (Ungar, 1960)
Individual testing
Counting bound
Combinatorial adaptive
Probabilistic adaptive
Aspect ratio, $A = T/n$
Prevalence, p
0
0.25
0.5
0.75
1

Rate

It can be useful to look at the “rate”

Rate $=$ bits learned per test $= n\, H(p)/T$

ratio of lower bound
to actual number of tests



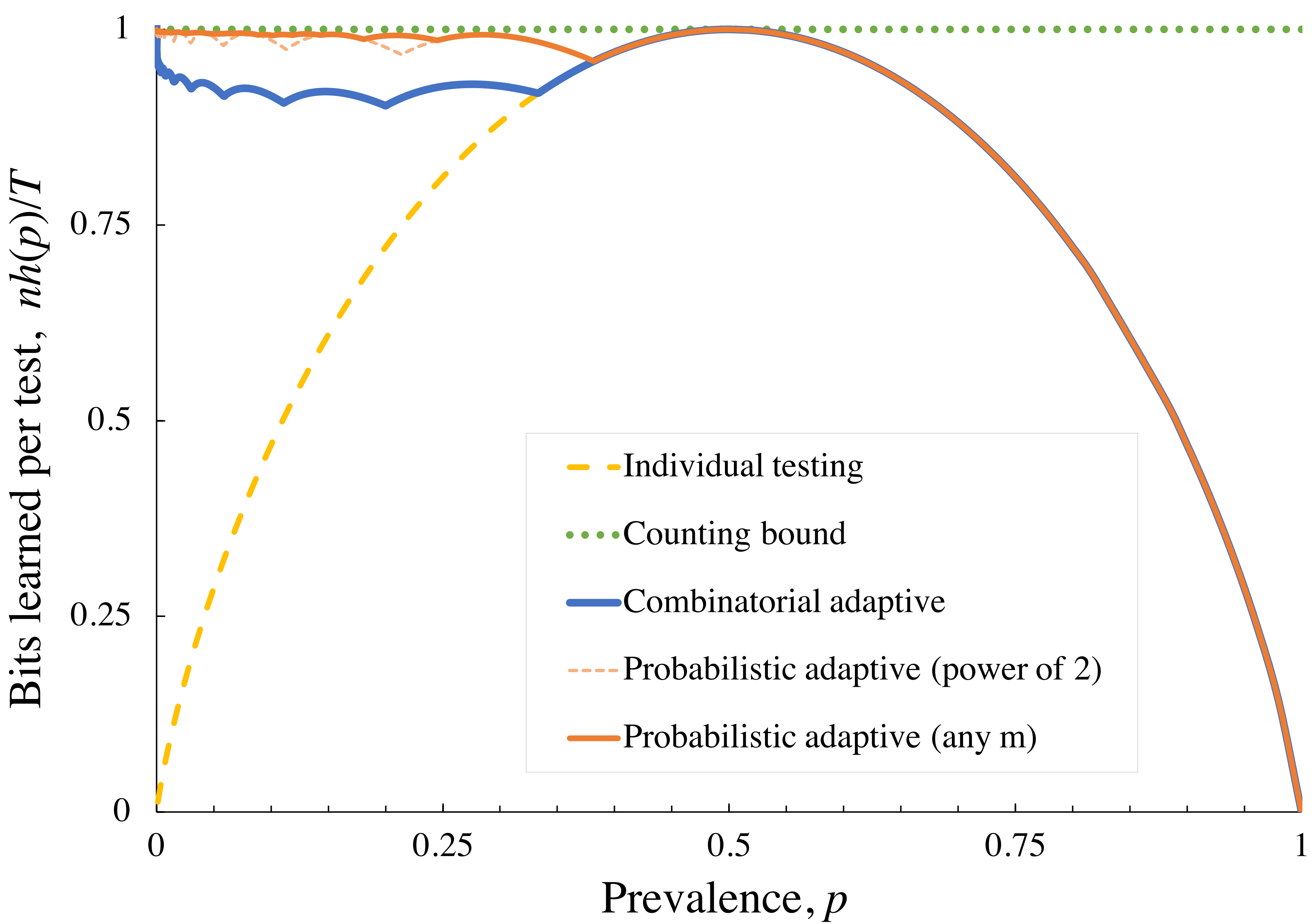
1
0.75
0.5
0.25
0
Bits learned per test, $nh(p)/T$
0
0.25
0.5
0.75
1
Prevalence, p
Individual testing
Counting bound
Combinatorial adaptive
Probabilistic adaptive (power of 2)

Improvement

We can sometimes do slightly better
for probabilistic testing
if we allow the set size m to not be a power of 2.

Use a Huffman tree for uniform probabilities
to organize the binary splitting.

(Zaman–Pippenger, 2016; Aldridge 2019)



Bits learned per test, nh(p)/T
1
0.75
0.5
0.25
0
Prevalence, p
0
0.25
0.5
0.75
1
Individual testing
Counting bound
Combinatorial adaptive
Probabilistic adaptive (power of 2)
Probabilistic adaptive (any m)

Open problem

Improve on these algorithms,
or show they are optimal.

Adaptive testing in the linear regime

We need to take greater care with "**error terms**".

We need to take care that **parameters that need to be integers** are integers.

There can be a bigger difference between average-case and worst-case behaviour.

Naïve algorithms can be optimal: try to find out when this is.

4
Nonadaptive group testing

Nonadaptive testing

The entire test design is fixed before we start
then all tests carried out in parallel

Combinatorial testing	**Probabilistic testing**
Exactly k defective items	Each item is independently defective with probability k/n
Must be certain to succeed whichever k items it is	Want to succeed with high probability as $n \to \infty$

Nonadaptive testing

The entire test design is fixed before we start
then all tests carried out in parallel

Combinatorial testing	**Probabilistic testing**
Exactly k defective items	Each item is independently defective with probability k/n
Must be certain to succeed whichever k items it is	Want to succeed with high probability as $n \to \infty$

Combinatorial nonadaptive

Individual testing is optimal for $k \gtrsim \sqrt{n}$

(D'yachkov–Rykov, 1982)

So the linear regime
is just the same as
some of the "denser" parts
of the sparse regime.

Nonadaptive testing

The entire test design is fixed before we start
then all tests carried out in parallel

Combinatorial testing	**Probabilistic testing**
Exactly k defective items	Each item is independently defective with probability k/n
Must be certain to succeed whichever k items it is	Want to succeed with high probability as $n \to \infty$

Probabilistic nonadaptive

In the **very sparse** (k constant) **regime**, we need

$$T = k \log_2 n$$

tests to succeed,

which matches the counting bound.

(Freidlina, 1975; Sebő, 1982)

Probabilistic nonadaptive

In the **very sparse** (k constant) **regime**, we need

$$T = k \log_2 n$$

tests to succeed,
which matches the counting bound.

Test items according to a "Bernoulli" design:
Each item is placed in each test independently
with probability $p = 1 - 2^{-1/k}$.

Probabilistic nonadaptive

In the **sparse** ($k = n^a$) **regime**, we need

$$T = \max\left\{k \log_2 \frac{n}{k},\ \frac{1}{\ln 2} k \log_2 k\right\}$$

tests to succeed,
which matches the counting bound for $a < 0.41$.

Atia–Saligrama, 2012
Chen–Che–Jaggi–Saligrama, 2011
Aldridge–Baldassini–Johnson, 2014
Aldridge–Baldassini–Gunderson, 2017
Scarlett–Cevher, 2017
Johnson–Aldridge–Scarlett, 2019
Coja-Oghlan–Gebhard–Hahn-Klimroth–Loick, 2019
Coja-Oghlan–Gebhard–Hahn-Klimroth–Loick, 2020

Probabilistic nonadaptive

In the **sparse** ($k = n^a$) **regime**, we need

$$T = \max\left\{k \log_2 \frac{n}{k},\ \frac{1}{\ln 2} k \log_2 k\right\}$$

tests to succeed,
which matches the counting bound for $a < 0.41$.

Test items according to a "constant tests-per-item" design:
Each item is placed in $L = (\ln 2)T/k$ tests
chosen uniformly and independently at random.

(Although for $a < 1/3$, the Bernoulli design is fine too.)

Probabilistic nonadaptive

In the **linear** ($k = pn$) **regime**, we need

$$T = n$$

tests to succeed,
so individual testing is optimal.
(Aldridge, 2018)

Probabilistic nonadaptive

Idea of the proof:

Supposed an item is “hidden”:
every test that item is in contains a(nother) defective item.

We can’t be sure whether the item is defective or nondefective.

Probabilistic nonadaptive

Idea of the proof:

Supposed an item is “hidden”:
every test that item is in contains a(nother) defective item.

We can’t be sure whether the item is defective or nondefective.

In the very sparse or sparse regimes,
we’re safe to guess it’s nondefective.

Probabilistic nonadaptive

Idea of the proof:

Supposed an item is “hidden”:
every test that item is in contains a(nother) defective item.

We can’t be sure whether the item is defective or nondefective.

In the very sparse or sparse regimes,
we’re safe to guess it’s nondefective.

But in the linear regime, we might guess wrongly.

Probabilistic nonadaptive

Idea of the proof:

If $T < n$,
then individual tests are wasted,
as they reduce the "tests per item" available.

So we can remove individual (or empty) tests,
and assume all tests have weight at least $w_t = 2$.

Probabilistic nonadaptive

Idea of the proof:

The probability item i is hidden in test t is

$$\mathbb{P}(i \text{ hidden in } t) = 1 - q^{w_t - 1}$$

The probability item i is hidden over all is

$$\begin{aligned}
\mathbb{P}(i \text{ hidden}) &= \mathbb{P}\left(\bigcup_{t \ni i} \{i \text{ hidden in } t\} \right) \\
&\geq \prod_{t \ni i} \mathbb{P}(i \text{ hidden in } t) \\
&= \prod_{t \ni i} (1 - q^{w_t - 1})
\end{aligned}$$

Probabilistic nonadaptive

Idea of the proof:

The probability item i is hidden in test t is

$$\mathbb{P}(i \text{ hidden in } t) = 1 - q^{w_t - 1}$$

The probability item i is hidden over all is

$$\mathbb{P}(i \text{ hidden}) = \mathbb{P}\left(\bigcup_{t \ni i} \{i \text{ hidden in } t\}\right)$$

$$\geq \prod_{t \ni i} \mathbb{P}(i \text{ hidden in } t)$$

$$= \prod_{t \ni i} (1 - q^{w_t - 1})$$

“Positively correlated”
FKG inequality

Probabilistic nonadaptive

Idea of the proof:

The probability item i is hidden in test t is

$$\mathbb{P}(i \text{ hidden in } t) = 1 - q^{w_t - 1}$$

The probability item i is hidden over all is

$$\mathbb{P}(i \text{ hidden}) = \mathbb{P}\left(\bigcup_{t \ni i} \{i \text{ hidden in } t\}\right)$$

$$\geq \prod_{t \ni i} \mathbb{P}(i \text{ hidden in } t)$$

$$= \prod_{t \ni i} (1 - q^{w_t - 1})$$

"Positively correlated"
FKG inequality

Probabilistic nonadaptive

Idea of the proof:

Check that the probability an item is hidden
averaged over the item i
is bounded away from 0.

Then there's some item
with positive probability of being hidden.

Then there's a positive probability
we guess wrongly whether or not it's defective.

Probabilistic nonadaptive

In the **linear** ($k = pn$) **regime**, with

$$T < n$$

the probability of success
is bounded away from 1...

(Aldridge, 2018)

…and in fact tends to 0.

(Heng–Scarlett, 2020)

Nonadaptive inference in the linear regime

Naïve algorithms can be optimal.

This is because we can't just assume an input is non-active if we lack evidence.

Probabilistic nonadaptive

This doesn’t mean that
nonadaptive group testing ideas
are not useful in the linear regime.

We can find many
defective and nondefective items
in fewer than n tests
(just not all of them)
(Heng–Scarlett, 2020)

5

In closing…

Things I didn't talk about

Group testing with noise

where the test results are sometimes wrong

Group testing with two (or more) stages

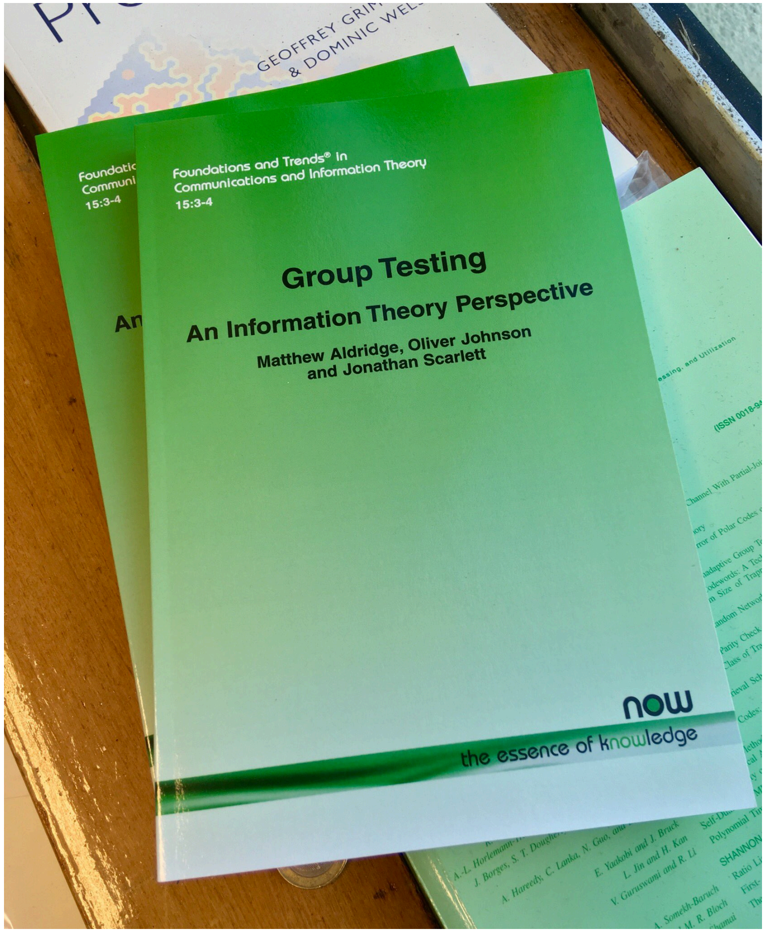
between adaptive and nonadaptive

Quantitative group testing:

We have a (possibly imperfect) measure
of how many defective items are in the test

M Aldridge, O Johnson and J Scarlett
Group Testing: An Information Theory Perspective
Foundations and Trends in Communications
and Information Theory, 2019

Preprint:
`arXiv:1902.06002`



Conclusions

Consider if the linear regime might be important for your inference problems.

Naïve sparsity-unaware algorithms (like individual testing) can be optimal.

Order-optimality is good, but look out for constants too.

“Error terms” often need more care.