Optimal Scheduling of Stochastically Independent Tests

Paul Kantor
(joint work with Endre Boros
Students: Noam Goldberg, Jonathan Word;
Randyn Bartholemew)

Outline

- Applications and fundamentals
- Detection at low budgets
- Variable thresholds and the ROC
- Using Multiple simultaneous tests
- Using Multiple tests sequentially problem
- Sequential: Linear programming
- Sequential: Dynamic Programming
- Open Problems

Required joke: a scholarly talk should include things understood by

- Everyone
- Students
- Graduate students
- Faculty
- Specialists
- Only the speaker
- No one

Not a joke:

9:00 in Paris, c'est 4:00am in New York

!!

Applications

- Testing, the practical application of science, is used in industry and medicine, in sports, and in security
- Strength of materials
- Indicators of disease
- The Lance Armstrong problem
- Passenger screening
- Nuclear threat detection

Fundamentals

- Tests are costly, and imperfect.
- Costs:
  - Capital costs: buy the machinery, train the workers
    - 1K detector – 5K stats good enough to determine 3-5sig change in count rate against background in 0.5 sec. Advanced Portal Monitor ~50x higher cost (300,000USD)
  - Operating costs: per case examined (bridge, patient, athlete, traveler, container, etc.)
- Imperfections:
  - Accuracy is less than 100%.
  - Requires two numbers to describe

Accuracy

- A simple binary test yields two results, which we can call Flag (F) and not (N).
- The accuracy involves two (stochastic) parameters

$$d = \Pr\{F \mid \text{target}\}$$
$$f = \Pr\{F \mid \text{not target}\}$$

- These are random not because of sensor behavior, but because of case variation

The value of knowledge

- The (expected) utility of taking one of the available actions

$$a \in A$$

- Depends on the “truth about the world”
  - *U(a,t).*
- *We want to maximize expected utility*
- *The value of an imperfect sensor is the increase in*
  - *EU(a,t|i,W) compared to EU(a,t|W prior).*

Applications

- This can be simplified to
  - Expected Utility(a=choice made) = + *f[U(a=wrong, t=not target)]+constant + (1-d)[U(a=wrong,t=target)+constant'] + constant"*
  - *We want to minimize EC(a)+C(tests)*
  - *The problem is that such calculations depend on the prior probbility, and on the unknowable large negative utility:U(a=wrong, t=not target)*
  - *We are still stuck*

Divide and conquer

- We combine cost of false alarms, which occur very often, with operating costs, and keep the cost of missed items as a separate consideration:
- Cost--> *f [U(a=wrong, t=not target)]+constant* + *(Cost of test)*
- *Value--> (1-d)U(a=wrong,t=target)*
- *At any given cost, we get the most value by maximizing d !*

We divide the problem

- Technical problem
  - for every value of the new cost, find that strategy producing the highest detection rate
  - present the (cost,detection) curve to a decision maker
- Policy problem
  - the decision maker decides what level of risk is acceptable, given competing demands on the budget.

The detection performance of any strategy

- *(f,d). These will depend on the sensors used, the sequence in which they are used, the decision rules, and the specific operating rules or (multiple) thresholds that are chose*
- *This is a purely technical computation, involving only sensor characteristics as they relate to the universe of threats.*

The cost-detection performance

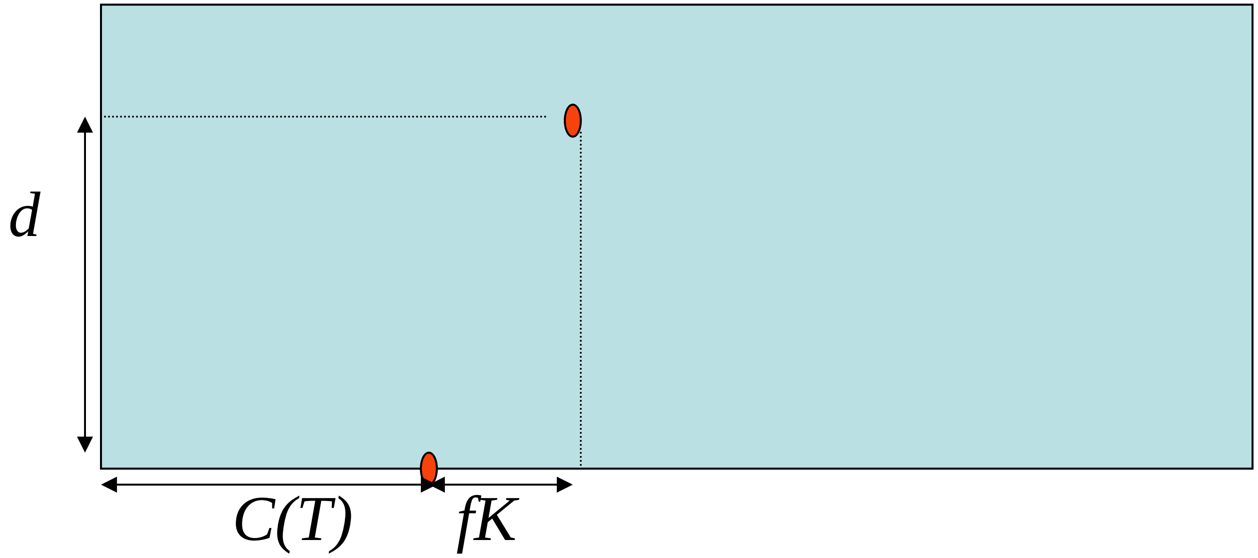
- At any operating point *(f,d), the operating cost (remember: not the cost of a disaster) is given by*
  - *$C(p)=C(Tests)+\pi dC(U)+(1-\pi)f(C(U)+C(I))$*
  - *C(p)= expected cost of policy p*
  - *C(Tests)= expected cost of the testing*
  - *C(U)= cost of unpacking (total inspection)*
  - *C(I) = cost of interruption to commerce*
  - *π = a priori probability of a threat (unkown)!.*

We are almost done

- *This relation still contains the unknown parameter π. However (this can be lifted if needed) we are going to use the fact that π<< 1, and neglect it compared to 1.*
- *Now the cost is just*
  - *C(p)=C(Tests(p)) +(1)f(p)(C(U)+C(I))*
  - *no troublesome Greek letters.* ☺
  - *we measure costs in units of C(U), so*
    - *C(p)=C(Tests(p)) +f(p)(1+K)*
    - *where K=C(I) interruption to commerce*

Detection at Low Budgets

- The typical cost-detection curve looks something like this. There is no detection until everything has been examined.



Detection is an *intensive* property

- Extensive property:
  - $V(X \cup Y)=V(X)+V(Y)$
  - example: volume of a gas
- Intensive property:
  - $T(X \cup Y)=|X|T(X)+|Y|T(Y)$
  - example: temperature of a gas. | | is a measure of the amount.

Intensive (continued)

- If the cases are divided into two groups, with sizes ….

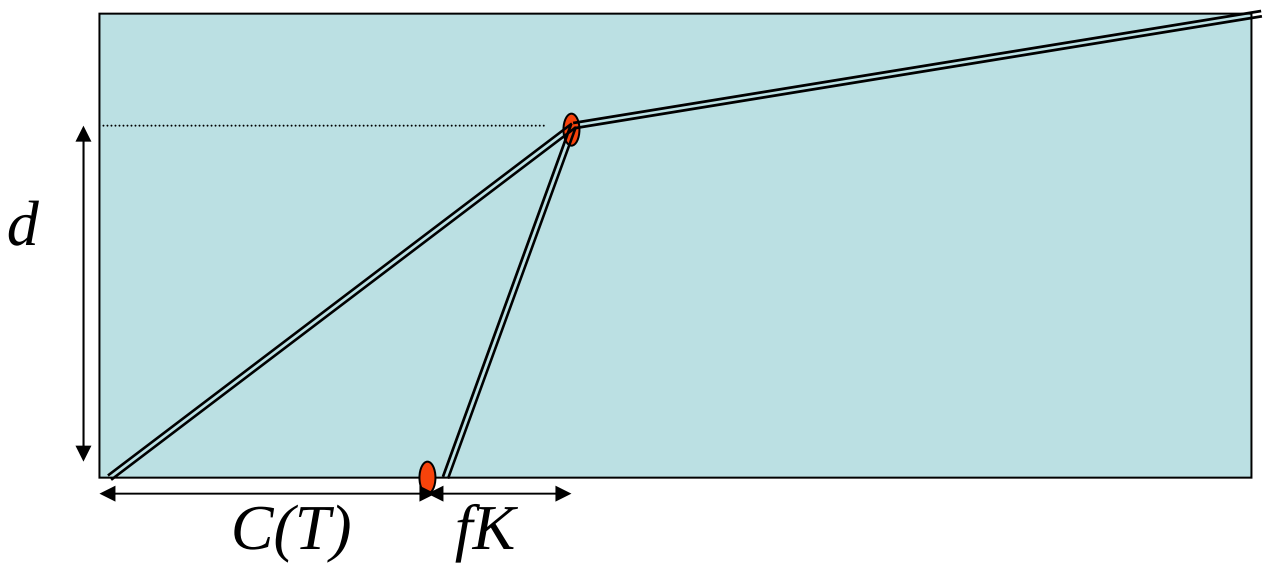
$$N_{1,} N_2 \ \ subjected\, to\ \ policies\ \ P_1, P_2$$

$$d(\mathbf{N,P}) = N_1 d(P_1) + N_2 d(P_2)$$

- And the same is true for decomposition into more than two sets
- The cost of inspection is also intensive.

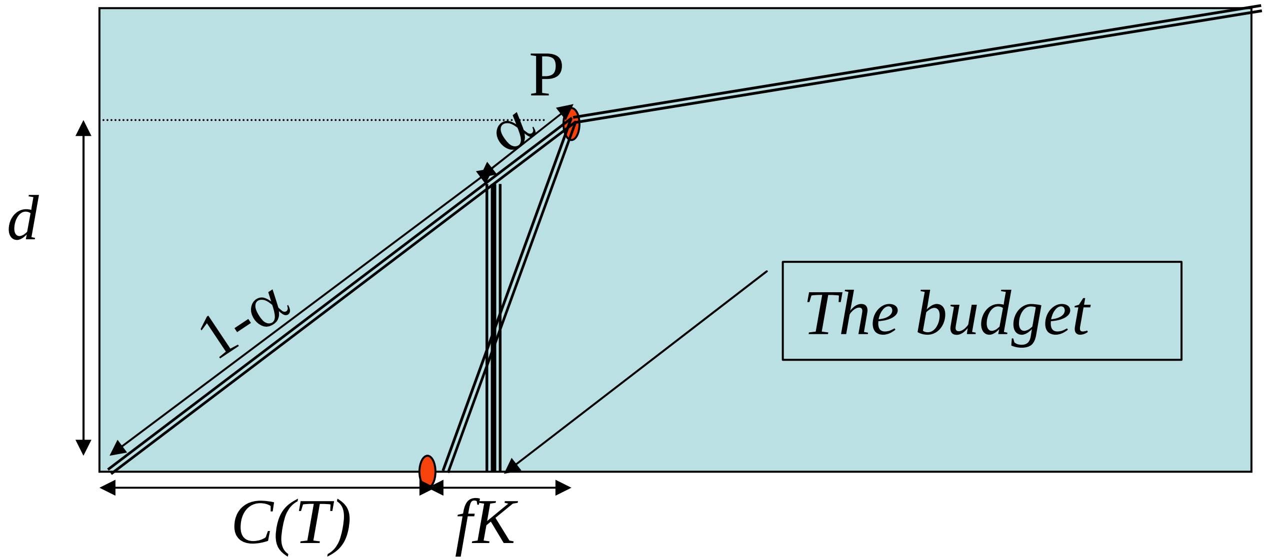
Convexity

- It follows that if any two (Cost, Detection) points are achievable, so is any point on the line connecting them.



For Low Budgets

- When there is not enough money to reach the point *P*, the optimal strategy is mixed, in the proportions needed to reach the budget, on the line from *0 to P.*



Screening Power Index

- If there are several available tests, with different cost-detection performance, in the regime of low budgets we can select among them based on a single number: Screening Power Index $G(P)=d(P)/C(p)$.

Multi-message tests

- We have so far assumed that a test either raises a flag *F, or does not.* In fact, a test may report any of serval messages, which we will label by *m*

$$m_T \in M_T$$

- for example, inspection of documents might yield the three results
  - {highly suspicious, somewhat suspicious, OK}

Ordering of labels

- These labels are placed in the "natural order".

$$\{\text{highly suspicious, somewhat suspicious, OK}\}$$

$$\Pr\{m_1 \mid t\} / \Pr\{m_2 \mid \neg t\} \text{ decreasing}$$

- This means that if the optimal strategy involves opening any of the cases with label *m,* we should also open all of the cases with label *m'*<*m.*

We assume this ordering

- If the labels were not in this order, we would simply rearrange them. Let *S(m) represent the set of cases receiving label m*
- We must now (we will relax this later) map the set of labels into the two actions {**Inspect**, **Release**}.
- Clearly, the optimal subsets matched to **Inspect** are of the form

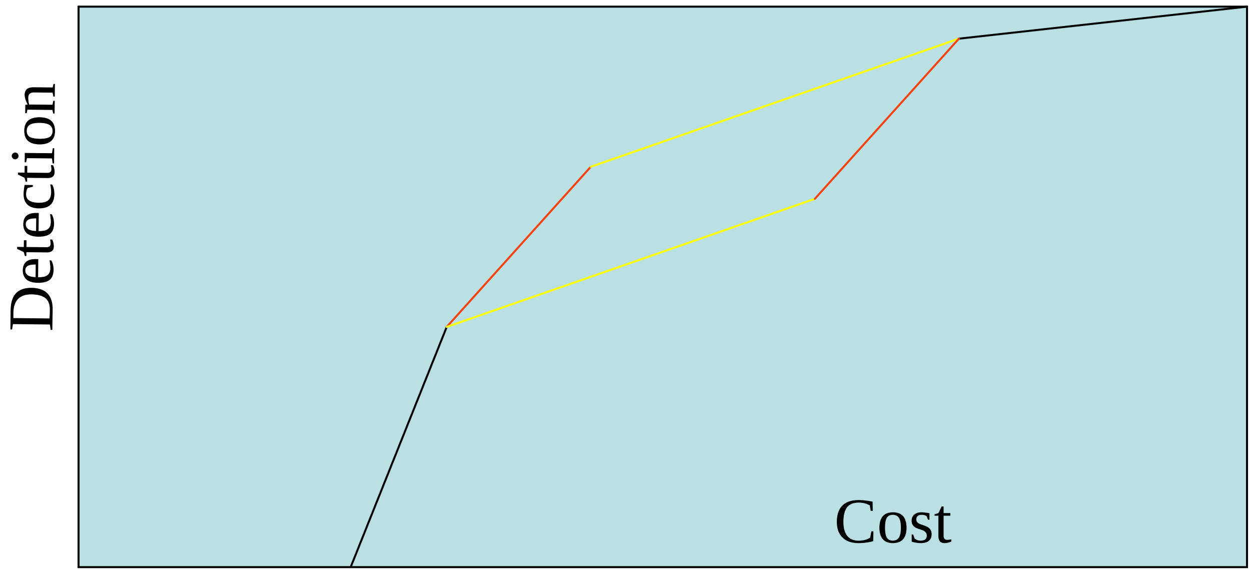
$$S(m^*) = \bigcup_{m<m^*} S(m)$$

First Monotonicity

- Rather than write out the equations, we can make it obvious as follows.
  - Each label generates some fraction of the detectable threats, and some other fraction of the harmless items. These line segments convert to segments in the C-d plot. And they must be taken in decreasing odrer of slope.

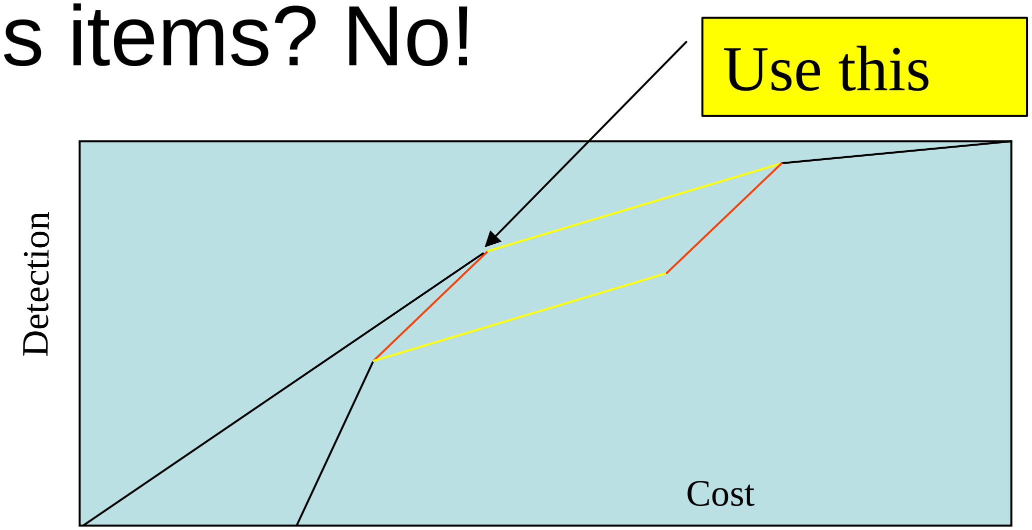
“Proof”

- If we took segment 3 (yellow) before segment 2 (red) the cost-detection curve would be dominated by the smarter choice.



How should we set the operating point

- For a given (low) budget, the best strategy is a mixture.
- But, should we always flag only the most suspicious items? No!



The screening power index

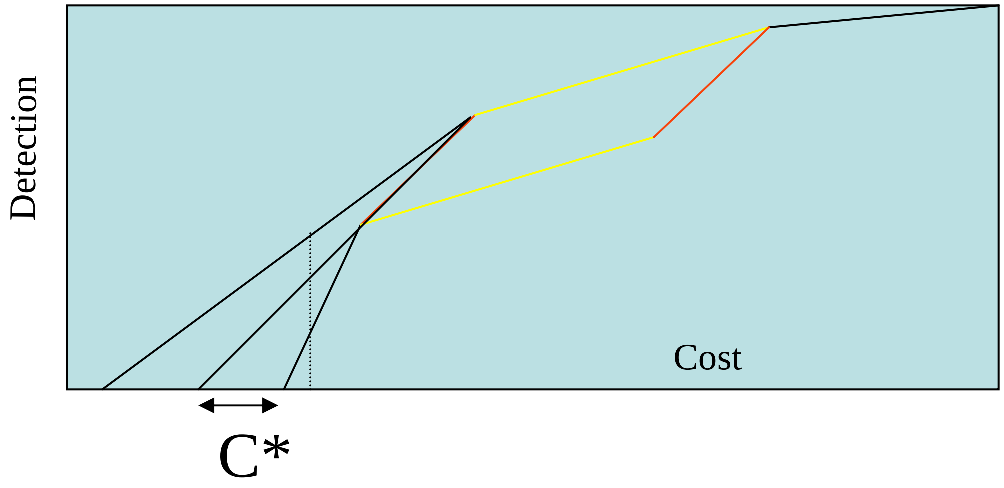
- For low budgets we define the screening power index *G* by the equation:

$$G(T) = \max_{trigger\ sets\ t\ of\ T} \frac{d(t)}{C(t)}$$

- If there are several possible strategies and a low enough budget, choose the strategy with the highest value of G.

Given performance, what is the "lowest cost for non-obvious solution"

- The portion of the plot to the right of C(T) does not depend on the cost of the test.
- If the cost is less than C*, opening only the most suspicious cases is opimal, with the budget determining the mixture. If the cost is above C*, then the mixture will include opening some of the less suspicious cases.



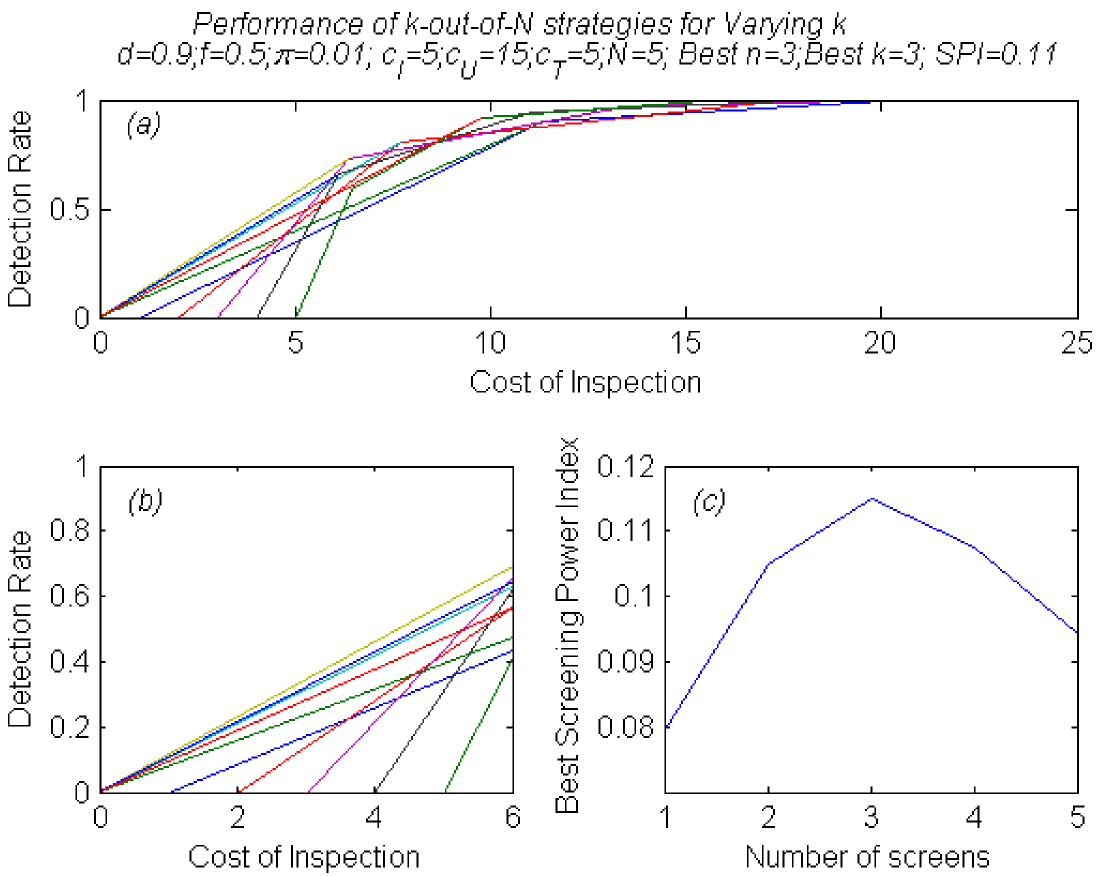
Multiple Simultaneous Tests

- Suppose we have a number of tests that can be conducted at the same time (various kinds of document check).
- For simplicity, suppose that each yields only a pair of labels or messages:
  - {More suspicious, Less suspicious}
- How many of the tests should we run?
- What should be our decision rule based on the results.

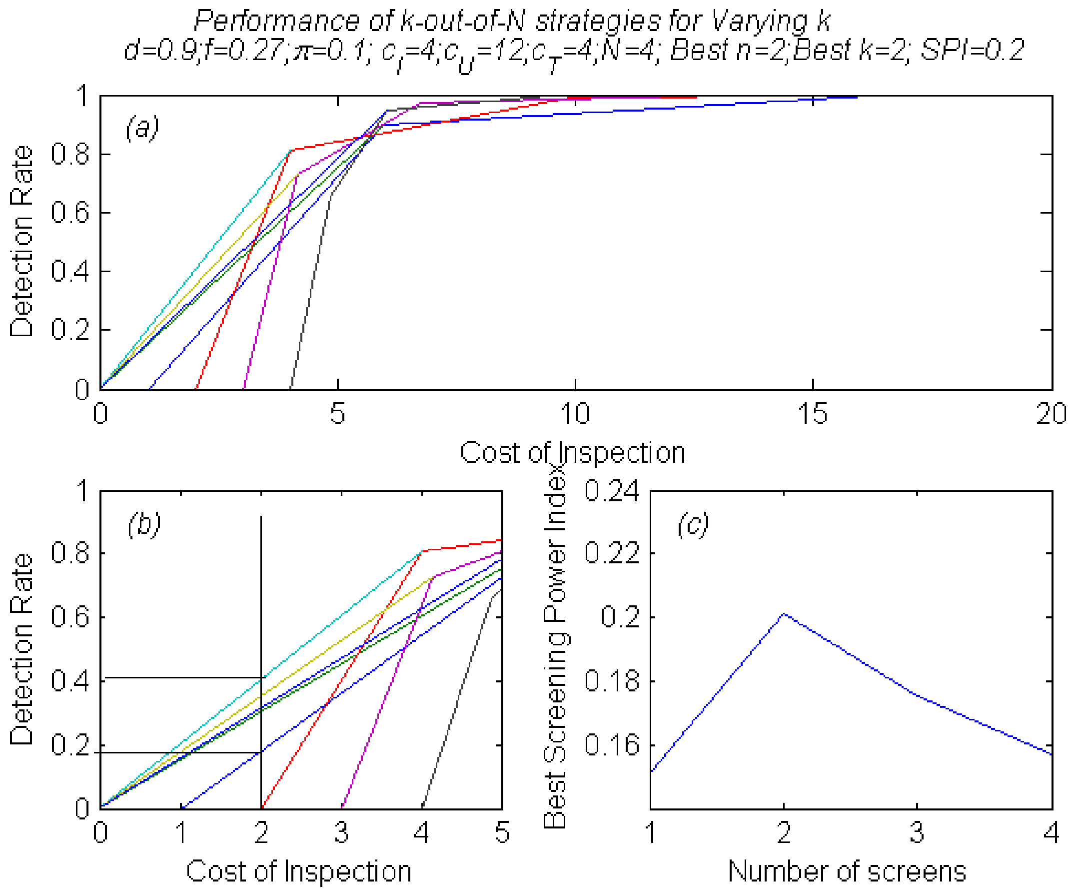
Simplifying assumptions

- 1. All of the tests have the same cost
- 2. All of the tests have the same *(f,d) parameters.*
- *The results are sometimes surprising*
- *To find the solution, we compute the (c-d) curves for each number of tests, and for each possible "k-out-of-n" rule (these are optimal when tests are identical).*

Example Results

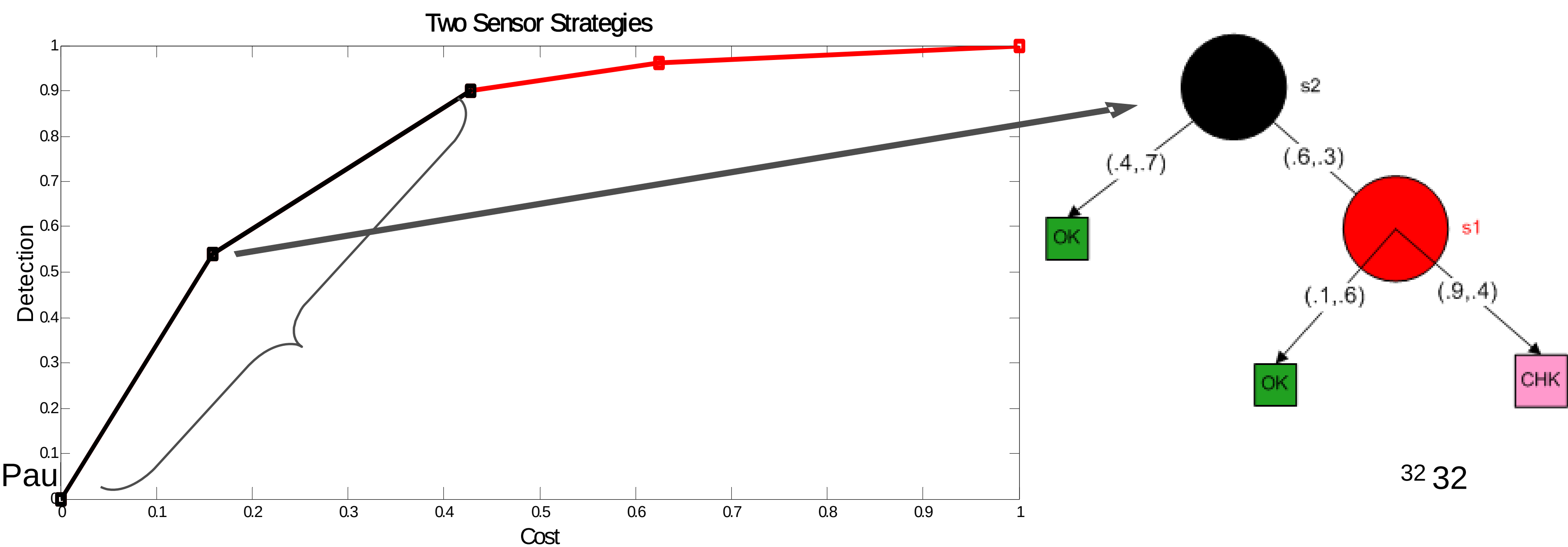


Example Results (2)



Sequencing of tests and domination

- When sequencing several sensors we may need to use a dominated sensor strategy.
- In this example the dominated, costly, sensor s_2 is optimally used as the root sensor



The general sequencing problem

- If something (a "case") has been examined with a set of sensors which I will call:

$$s_{i(1)}, s_{i(2)}, \ldots s_{i(k)}$$

- Yielding a set of readings or labels

$$m^{i(1)}{}_1, m^{i(2)}{}_2, \ldots m^{i(k)}{}_k$$

- then we know something about the odds that it is harmful

The odds ratio

- The a priori odds that this case is a target have been increased by the Bayes factor:

$$\frac{\Pr\{m^{i(1)}{}_1, m^{i(2)}{}_2, \ldots m^{i(k)}{}_k \mid t\}}{\Pr\{m^{i(1)}{}_1, m^{i(2)}{}_2, \ldots m^{i(k)}{}_k \mid \neg t\}} =$$

$$\prod_{j \le k} \frac{\Pr\{m^{i(j)}{}_j \mid t\}}{\Pr\{m^{i(j)}{}_j \neg t\}}$$

The path history

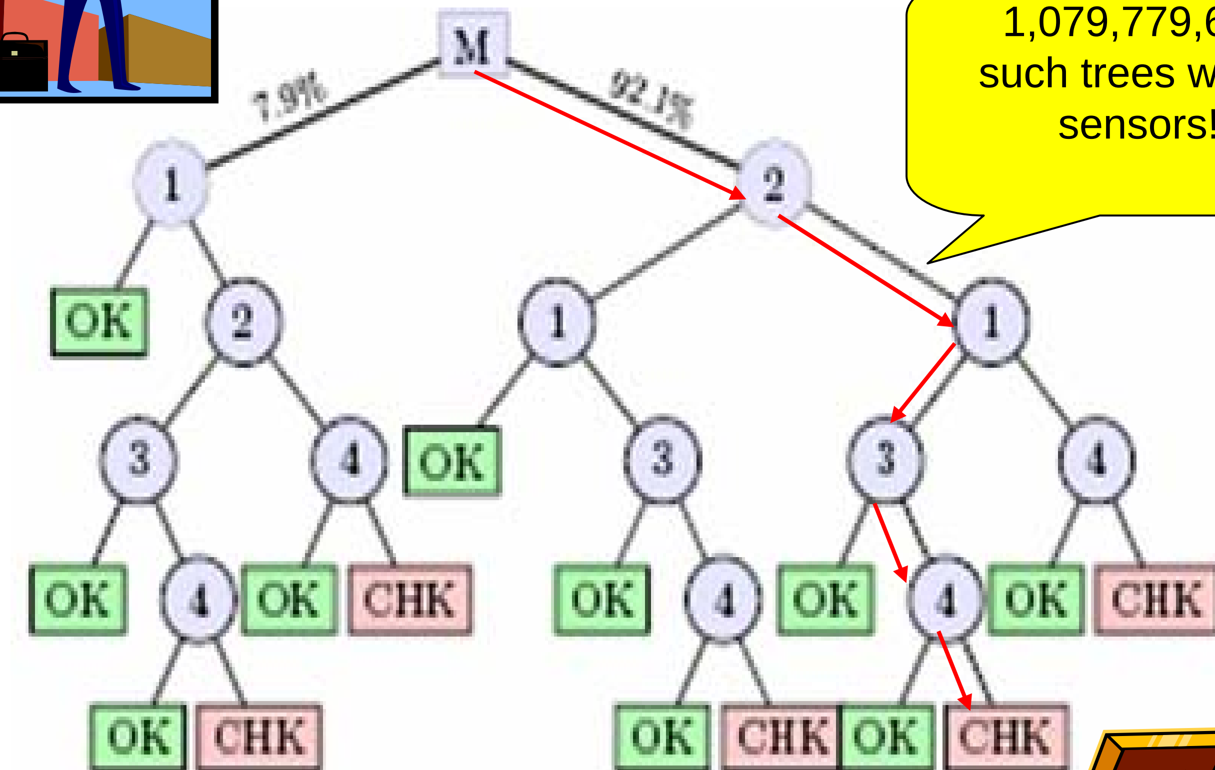
- The cases have an odds ratio completely determined by their “path history” -- which sensors they have met, and what readings resulted. So the odds ratio, which we will call Lambda, depends only on the set of labels

$$\Lambda(I) = \Lambda\left(m^{i(1)}{}_1, m^{i(2)}{}_2, \ldots m^{i(k)}{}_k\right)$$

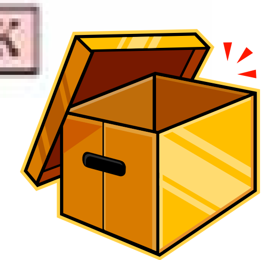
The Linear Program

- Whatever set of policies we establish, they will define a tree which can be represented this way (just a moment)
- and corresponding to each path, there will be a certain fraction of the targets, and a certain fraction of the "not-targets" that follow the path. Label the path η and the fraction following it $y(\eta)$

An inspection policy

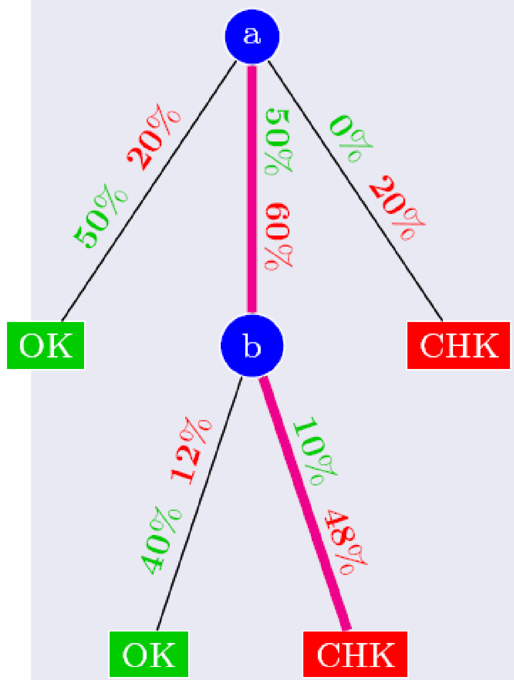


1,079,779,602 such trees with 4 sensors!



Linear Programming Model for Container Inspection

- Every decision tree can be viewed as a collection of histories with corresponding y_η values:



$$\eta(1) = \{(a,1), OK\} \qquad y_{\eta(1)} = 0.5$$
$$\eta(2) = \{(a,2),(b,1), OK\} \qquad y_{\eta(2)} = 0.4$$
$$\eta(3) = \{(a,2),(b,2), CHK\} \qquad y_{\eta(3)} = 0.1$$
$$\eta(4) = \{(a,3), CHK\} \qquad y_{\eta(4)} = 0.0$$

Linear Programming Model for Container Inspection

- Every decision tree can be viewed as a collection of histories with corresponding y_η values:
- Let Ξ be the set of all possible histories, and y_η denote the *fraction of good containers* we plan to test along history η (for all $\eta \in \Xi$).
- Detection rate, unit inspection cost, sensor utilization, time delay, etc., are all linear functions of $\mathbf{y} = (y_\eta | \eta \in \Xi)$.
- **Which vectors $\mathbf{y} = (y_\eta | \eta \in \Xi)$ correspond to (mixed) strategies?**

Let $\mathbf{y} = (y_\eta | \eta \in \boldsymbol{\Xi})$, and consider the polyhedron

$$P = \left\{ y \;\middle|\; \begin{array}{rcl} \sum_{\substack{\eta \in \boldsymbol{\Xi} \\ (\nu,(s,j)) \in \boldsymbol{\Pi}(\eta)}} y_\eta & = & \gamma(s,j) \sum_{\substack{\eta \in \boldsymbol{\Xi} \\ (\nu,(s,*)) \in \boldsymbol{\Pi}(\eta)}} y_\eta \quad \forall (\nu,(s,j)) \in \boldsymbol{\Pi} \\ \sum_{\eta \in \boldsymbol{\Xi}} y_\pi & = & 1 \\ & y_\eta \geq 0 & \forall \eta \in \boldsymbol{\Xi} \end{array} \right\}$$

Theorem

Vertices of P correspond to decision trees; $y \in P$ correspond to mixed strategies.

Given any particular budget, and any other capacity constraints, this problem can be solved using COTS LP solvers. It is a large problem, but smaller than the non-linear search used previously. The optimal solution will be a mixture of at most J+1 pure solutions, where J is the number of linear constraints.

Linear Programming: summary

- This approach, in which the possible solutions are found as the vertices of a polyhedron in the space of all possible paths through all possible trees, is a substantial improvement over enumerating all trees and doing a non-linear search over thresholds.

But…

- As a Linear programming problem, it takes the form:

$$\max \; d(Strategy)$$
$$subject\ to: C(Strategy) = Budget$$

- We have to know the budget to solve the problem.
- Is there a simpler way?

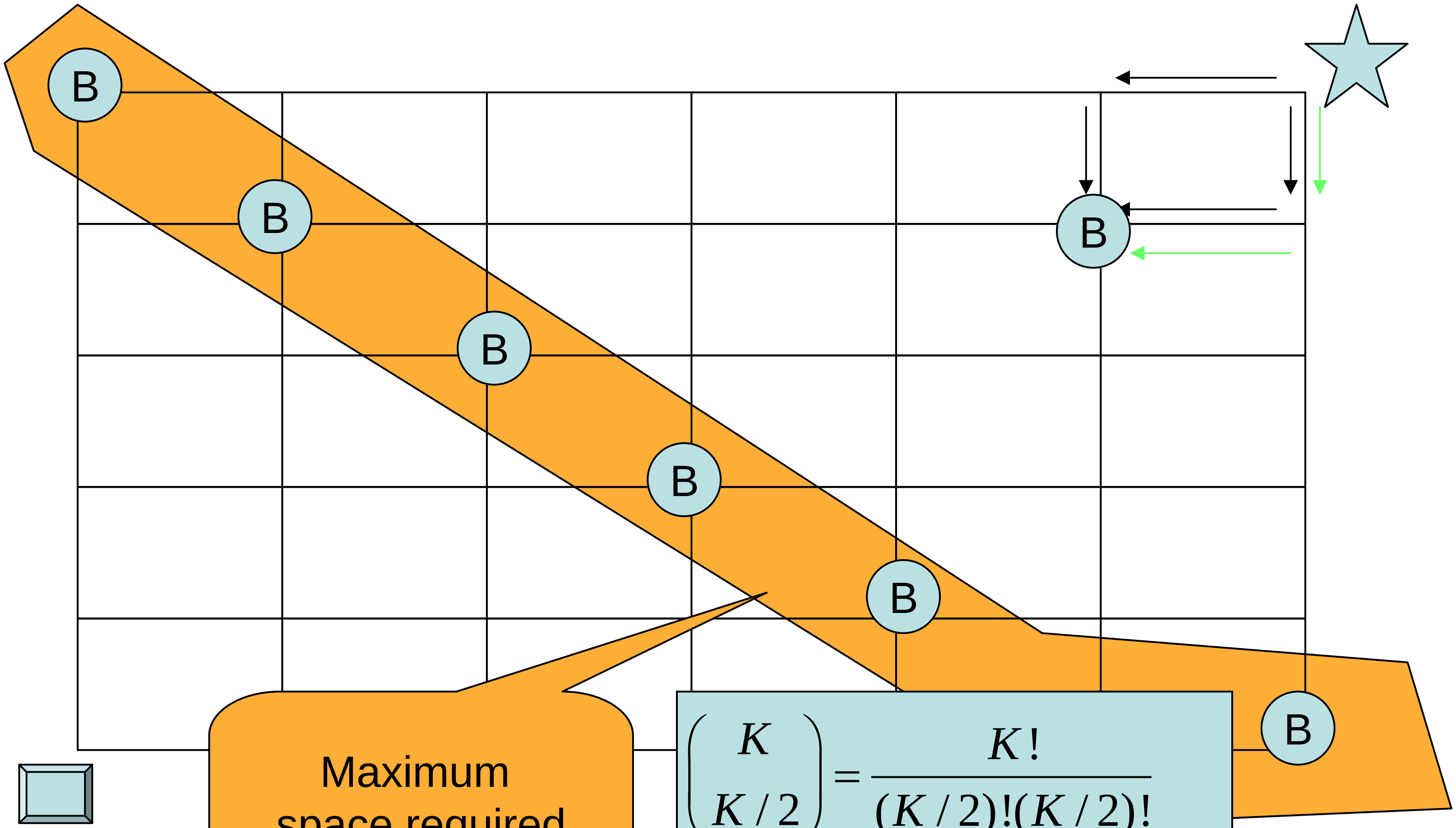
Towards dynamic programming

- When a case has any specific history, it also has available various policies, each of which has its own detection and cost parameters. So we should be able to make an optimal assignment of the case, to one of the available policies, which must only use the sensors not appearing in the path *-- remember, we assume randomness is in targets, not sensors.*

An insight

- For any particular combination of sensors (a subset of all of them) there is some best detection strategy (cost-detection curve).
- We can consider all the subsets with, e.g. 3 sensors. Find the best strategy for each subset.
- Consider, in turn, using any of the others to do a preliminary triage
- Find the best mixture of strategies
- And then drop all the dominated strategies

Dynamic Programming



Maximum space required
K=20: 184,756

$$\binom{K}{K/2} = \frac{K!}{(K/2)!(K/2)!}$$

$For\ K = 20\ this\ is\ 184{,}756$

Dynamic programming overview

- Dynamic programming finds ***the entire cost-detection curve at once.***
- Basic Facts:
  - Every optimal strategy is a mixture of at most 2 pure strategies.
  - The *efficient frontier* is a piecewise linear curve in cost-detection space consisting of the optimal strategies for each budget value.
- Solution:
  - Find curve by enumerating vertices – efficient *pure* strategies
  - Use cost-detection dominance when possible
  - The frontier for *k+1* sensors is constructed using all (already computed) subsets containing *k* sensors different from the one added.
  - We call adding another sensor above an existing set of (pure) strategies, the "*sensor prefixing problem*"

The key to prefixing

- Each label or message coming out of the top sensor represents a particular odds ratio:

$$\Lambda(top) = \Lambda\left(m^{i(top)}{}_{1_{top}}\right)$$

- And every policy *P* to which it might be prefixed has a particular ratio *d(P)/C(P)*
- *to assign each m to some P Sort decreasing by*

$$\Lambda(top, m) * d(P) / C(P)$$

The sensor prefixing problem

- Given a set of available testing strategies *(C_1,D_1), …, (C_N,D_N)*, in cost-detection space, and a sensor with a set of bins characterized by *(b_1,g_1), …, (b_T, g_T)*: *b_m=Prob{label=m|t)*
  - assign bins to strategies and maximize detection?
  - every bin assigned
  - For each budget (C): a new special case of a
  - Linear Multiple Choice Knapsack problem (Zemel 1980).
  - Can be solved ("greedy" algorithm) *O(NTlgT+NlgN)* - for all values of budget *M*
  - testing strategies are sorted; solve in *O(NT lg T)*

$$D^{(new)}(M) = \max \sum_{i=1,\, j=1}^{i=T,\, j=N} b_i D_j x_{ij}$$

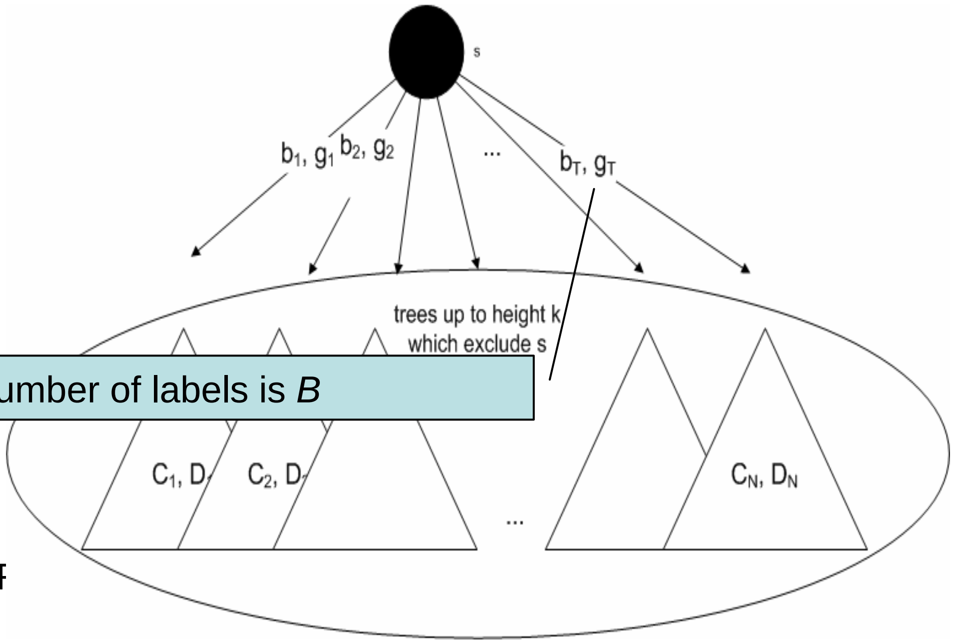
$$s.t.$$

$$\sum_{i,j} g_i C_j x_{ij} \le M$$

$$\sum_j x_{ij} = 1 \qquad for \quad e$$

$$0 \le x_{ij} \le 1 \qquad for \quad each \quad i, j$$

The number of labels is *B*

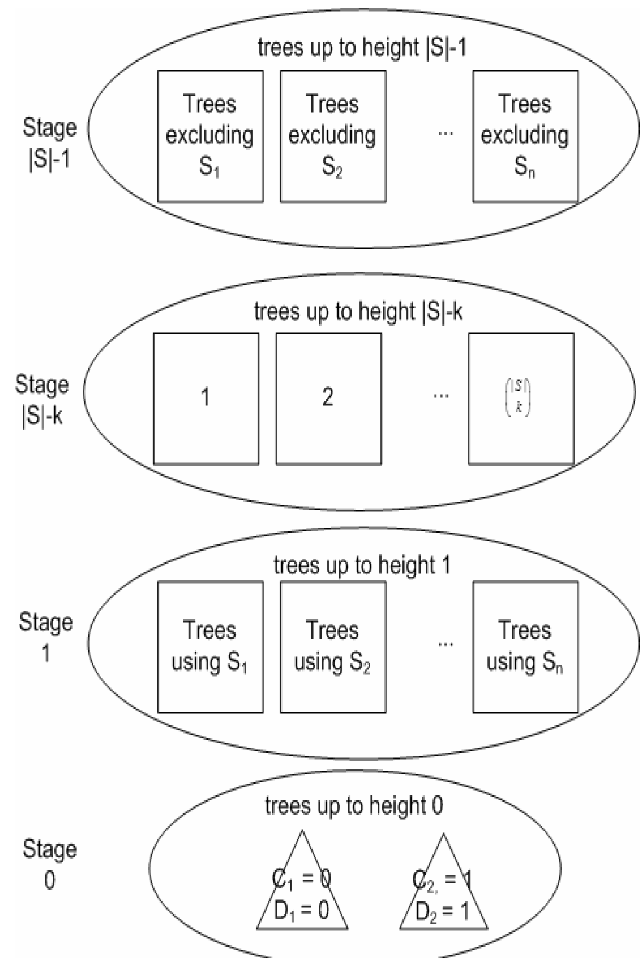


Dynamic programming formulation: the Math

$$f^{(k+1)}(S) = convhull\left(\bigcup_{s \in S} sensorinsertion(s, f^{(k)}(S - \{s\}))\right) \cup \{(0,0),(1,1)\}$$

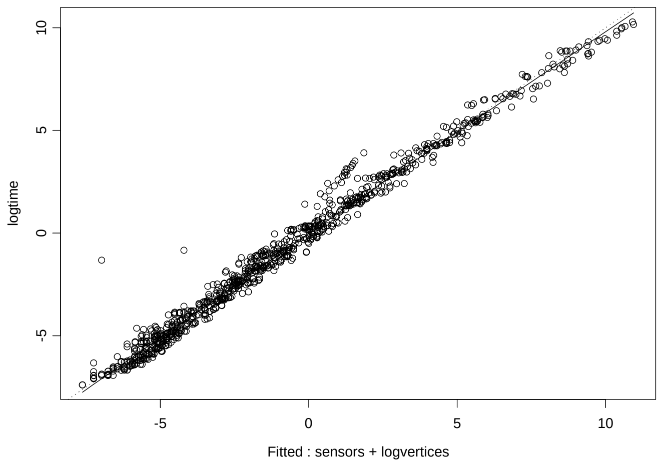
$$f^{(0)}(S) = \{(0,0),(1,1)\}$$

- Let $f^{(k)}(S)$ be the set of efficient frontier vertices of height *k* using sensor set *S*
- Stages correspond to the height of the strategy trees
- No more than in total $2^{|S|}$ possible subsets of sensors along a path



Representative Results

- Parametric Models:Running time depends on number of vertices:



$$T = 2^{-10.52} 2^{1.12(Sensors)} Vertices^{.65}$$

$$R^2 = 98.5\%$$

Number of vertices

- Depends, empirically on the number of sensors and the number of branches or bins:

$$Vertices = 1.46\left(1.37^{Sensors}\right)\left(1.55^{Bins}\right)$$

$$R^2 = 44\%$$

- *This quality would be great for social science; we want more*.

$$Vertices = 6.96 * (.75)^{Sensors} * (.75)^{Bins} (1.3)^{Bins \times Sensors}$$

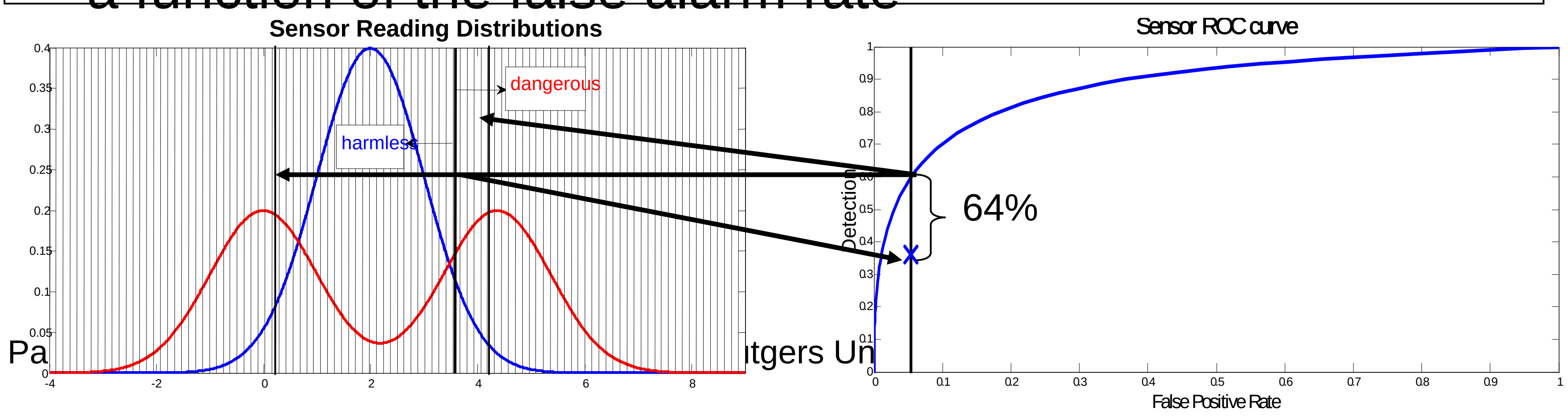
$$R^2 = 81\%$$

Why do we care about running time?

- For some problems the test yields a continuous parameter, such as total counts of a specific type of radiation. The ROC figure is then a smooth curve. But our calculation requires that it be made discrete. The tradeoff is between the accuracy with which we approximate the curve, and the time+space cost of the calculation.

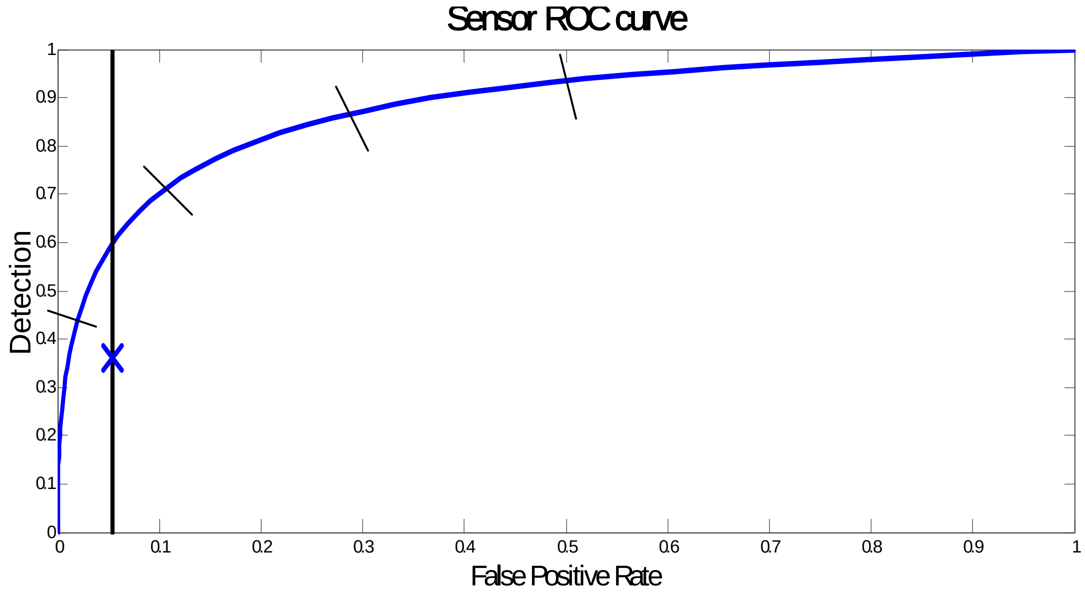
"Bins": From signal space, to Receiver Operating Characteristic (ROC)

- The naive approach is to assign a fixed number of bins in the space of scores.
- The distributions in the space of scores define an "odds ratio"
- The best detection for a given rate of false alarms is found by selecting regions of the space of scores, in decreasing order of the odds ratio (Neyman-Pearson)
- The ROC curve plots the resulting *d(f)* - detection rate as a function of the false alarm rate



Bins in ROC space

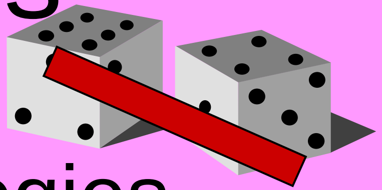
- We have been able to show that it is more effective to define bins in ROC space, and then translate them back into signal space.



Summary

- With this approach we have been able to speed up the calculations (versus non-linear grid search) by 6 to 9 orders of magnitude (estimated, of course)
- I hope I have piqued your interest in this class of problems, as
- By no means have we exhausted the interesting and important questions that may be explored.
- Let’s look at a few of them

Open Social Problems



- Can decision makers accept strategies that involve some level of randomness?
- The expected performance of such strategies is provably better, but
- The political consequences of a missed threat would be much worse since the strategy could be criticized as “random” – an avoidance of responsibility.

Open Social Problems

- *Lobbying based on K*
  - *we have tried to separate the political from the technical, but vendors of tests point to K, and argue that because it is so large, the goal is to minimize K. This is not wrong, but it does not attend to the primary goal of increasing d. It is, in a sense, a "peacetime" goal, rather than a "wartime" goal.*

Open technical problems

- *How to solve for threats where π is O(1)?*
  - *We speculate that the DP approach will generalize, but that the state space will now have two parameters: B and π*
- *How to control the computational burden when the reading(s) from a sensor are continuous?*
  - *We believe that the errors of the DP algorithm can be strongly bounded, but the proofs have not revealed themselves to us yet.*

Open technical problems (2)

- *What are the effects of randomness*
  - *We have been examining only the expected values of the detection, and of the cost. But these are both random variables. What are the policy implications of observing a lower d than is expected? What happens if f is higher than expected and we run out of money?*
  - *This problem was examined by M. Maschler, in the context of nuclear disarmament*

Open technical problems (3)

- *How to deal with the case of stochastically dependent sensors:*
  - *Prob{L(s), L(s')|t}≠ Prob{L(s)|t} Prob{L(s')|t}*
  - *in this case, the backward algorithm does not seem to work. There is then a hard problem of reducing the computation to tractable size.*
  - *We know the LP approach will work -- but even a supercomputer will not be adequate **and** the precise nature of the interrelations is not known.*

Open technical problems (4)

- *Real data are spectral profiles, not single readings. The randomness is compounded by the fact that sensors, at any energy bin, are seeing Poisson variates. It is all computable, but one needs to "put the machinery into a shrink-wrapped tool" for the decision makers, since they cannot share the data with us.*

Open technical problems (5)

- There are, in reality, multiple threats (highly enriched uranium, plutonium, cocaine). The correct secondary action will depend on what kind of threat is indicated by the primary test. How is this to be modeled?

Open technical problems (6)

- *The problem is embedded in a game (Inspector game) and the opponent can allocate resources to attacking through various channels, while we must allocate our resources to defending the several channels.*
  - *Maschler (1966). Leader (Stuckelberg) game. We are not harmed by the fact that we must announce. Is that true here?*

Merci Beaucoup

- Thanks to our sponsors:
  - US DHS DNDO #CBET-0735910
  - US DHS DyDAn Center
  - US ONR Port Security
- I will do my best to answer questions. ☺

To read more

- Testimony of Vayl Oxford; Director US DNDO.
  - http://homeland.house.gov/SiteDocuments/20080305142759-83992.pdf
- Linear model; no independence assumptions;
  - http://rutcor.rutgers.edu/pub/rrr/reports2006/26_2006.pdf
- Screening Power Index (low budgets only)
  - http://rutcor.rutgers.edu/pub/rrr/reports2007/26_2007.pdf
- Dynamic Programming. Stochastic Independence, π=0.
  - http://rutcor.rutgers.edu/pub/rrr/reports2008/14_2008.pdf