Measuring Intrusion-Detection Systems

Roy A. Maxion

Carnegie Mellon University
School of Computer Science

Presented to

The First International Workshop on Recent Advances in Intrusion Detection

(RAID-98)

14-16 September 1998

Louvain-la-Neuve, Belgium

Overview

- Basic tenets of measurement
- To measure, or not to measure?
- Dependability cases
- A measurement plan
- Quantitative figures of merit
- Pitfalls
    - Measurement
    - Experimentation
    - Simulation
    - Evaluation
- Example measurement experiments
    - Detecting anomalies in network traffic
    - Synthetic anomaly detection
- Progress in intrusion detection
- Conclusion

What does it mean to measure?

Measure:

- to appraise by a certain standard or rule, or by comparison with something else
- to ascertain the different dimensions of a body
- to gauge the abilities or character of
- to estimate the amount, duration, value, etc. of something by comparison with some standard
- a standard; a criterion; a test

The assignment of numerals to objects or events according to a rule. - Stevens 1959

The process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to characterize the attributes by clearly defined rules. - Fenton 1991, 1995

Why measure?

1. To characterize
    - Understand processes, artifacts, resources & environments
    - Establish baselines for comparisons
    - Comprehend relationships among measures
2. To evaluate
    - Determine performance levels
    - Assess impacts of technologies, environments & variabilities
3. To predict
    - Extrapolate trends
    - Assess risk
    - Establish achievable quality goals (e.g., reliability, availability, speed, accuracy), costs and schedules
4. To improve
    - Identify obstacles, root causes & inefficiencies

Adapted from Park et al. 1996: Goal-Driven Software Measurement.

Uses of measurement

- Provide focus
- Determine how well something works
- Set direction for improvements
- Reduce variability (and increase confidence)
- Expose opportunities, not failures

To measure, or not to measure?

Some issues:

- Producing dependable mission-critical systems.
- Defending in court litigations
- Demonstrating due diligence
- Differentiating among products (truth in advertising)
- Performing sensitivity analyses
- Establishing performance boundaries
- Estimating/planning realistically
- Benchmarking against other artifacts
- Determining progress
- Assessing quality
- Recognizing improvement opportunities
- Recognizing achievements
- Meeting level-of-service obligations
- Assembling dependability cases

Dependability

Dependability: that quality of a delivered service such that reliance can justifiably be placed on the service.

Reliability: The probability that the system will survive a specified time interval.

Availability: The fraction of time that a system is available to perform its intended mission as specified.

Dependability subsumes the entire "ability" family ... reliability, availability, usability, maintainability, diagnosability, testability, learnability, etc.

Dependability case

A documented body of evidence that provides a convincing and valid argument that a system is adequately dependable for a given application in a given environment.

A clear, defensible, comprehensive and convincing argument ... aimed at identifying the risks (of failure) inherent in operating a system, demonstrating that the operating risks are fully understood, that they have been reduced to an acceptable level and are properly managed.

Implementing a dependability case requires:

- making an explicit set of claims about the system
- providing a systematic structure for marshalling the evidence supporting those claims
- providing a set of arguments linking the claims to the evidence
- making clear the assumptions and judgments underlying those arguments
- provide for different viewpoints and levels of detail

Some measurement-related questions ....

- More, bigger, smaller, faster, cheaper ...
- How big is it?
- How much is there?
- How many component?
- How fast is it?
- How long does it take?
- How much does it cost?
- Is the process stable?
- How is it performing now? (baseline)
- What limits our capability?
- What determines quality?
- What determines success?
- What things can we control?
- What do our customers want?
- What limits our performance?
- What could go wrong?
- What might signal early warnings?
- How will we know any of these things?

Figures of merit for intrusion-detection systems

- Speed
- Accuracy (hit, miss, false alarm)
- Response latency
- Overhead
- Noise
- Stimulus load
- Variability
- Usability

When these figures are used, they must be dependable.

Measurement pitfalls

- Poor definitions of metrics and procedures
- Operationalizing measures
- Validity
- Observability
- Controllability
- Fidelity
- Variability
- Interrater reliability
- Repeatability
- Ground truth
- Sampling
- Environment
- Representativeness
- Nonstationarity
- Overgeneralizing conclusions
- Lack of shared dataset or benchmark
- Lack of a baseline/benchmark

What constitutes a good definition?

- Describes the essence or nature of the thing defined, not its accidental properties.
- Gives the genus/differentia of that which is defined.
- Not defined by synonym or through metaphor.
- Written in very clear, concise, grammatical English.
- Using as few base terms as possible, and using these in an accepted dictionary sense (which ideally is quoted in the document).
- Ideally, all the rest of the definition should be easily translatable into set-theoretic terms, even if this is not done explicitly.
- Built up through a sensibly-chosen hierarchy of definitions.
- Well-illustrated with helpful examples, that are clearly differentiated from the definition text.
- Unambiguous (high interrater agreement).

Making a definition operational

Operational definitions tell users how data are collected. If you change the method or definition, you change the result. When users of data do not know how the data were collected, they easily make invalid assumptions, leading to incorrect interpretations, improper analyses, and erroneous conclusions.

Operational definitions must satisfy two important criteria:

1. Communication - Will others know what has been measured, how it was measured, and what has been included and excluded?
2. Repeatability - Could others, armed with the definition, repeat the measurements and get essentially the same results (within experimental error)?

If definitions aren’t made operational, you will essentially be saying, "I don’t care how you do it - you make the decisions."

Example: Measuring the height of school children.

Adapted from Park et al. 1996: Goal-Driven Software Measurement.

Examples of murky definitions (in ID)

- Misuse-based
- Host-based
- Network-based
- Availability
- Reliability
- Fault tolerance
- Resilience
- Real time
- Abuse
- Stressful conditions
- Robustness
- Intrusion

Network anomaly-detection experiment

Goal: Detect and diagnose injected faults/anomalies, discovering a single decision rule that covers multiple environments.

- Five fault/anomaly types
- Five characteristically different live network environments
- One fault/anomaly injection per hour
- 600 total injections over five weeks

Evaluation pitfalls

- No goals
- Biased goals
- Unsystematic approach
- Misunderstanding the problem
- Incorrect performance metrics
- Unrepresentative workload
- Wrong evaluation technique
- Overlooking important parameters
- Ignoring significant factors
- Inappropriate experimental design
- Inappropriate level of detail
- Inappropriate analysis
- No sensitivity analysis
- Ignoring errors in input
- Improper treatment of outliers
- Assuming no changes in environment
- Ignoring variability
- Analyses didn’t help shape decisions
- Omitting assumptions and limitations

Why these faults?

- All of the injected faults occur naturally in Ethernet environments.
- They are representative of a wide range of problems on an Ethernet network.
    - Network paging is a problem of resource misuse, either intentional or not.
    - Broadcast storms often result from incorrect implementations of network protocols.
    - Bad-frame checks can result from hardware failure.

Frequency of Occurrence:

Runt storm 28%

Network paging 12%

Bad frame check 0%

Jabber 18%

Broadcast storm 13%

Other 29%

Low- and high-performance machines

Machines typical of each type are:

Low Performance

- Sun-3, IBM RT, NeXT, Encore Multimax, microVAX/VAX, Motorola MC68xxx, Sequent Balance, Gateway, Toshiba 5200, IBM L40SX, Intel i386, Omron, Ariel, Macintosh, Olivetti M380, Kinetics, Sony NWS-1250, TI Explorer.

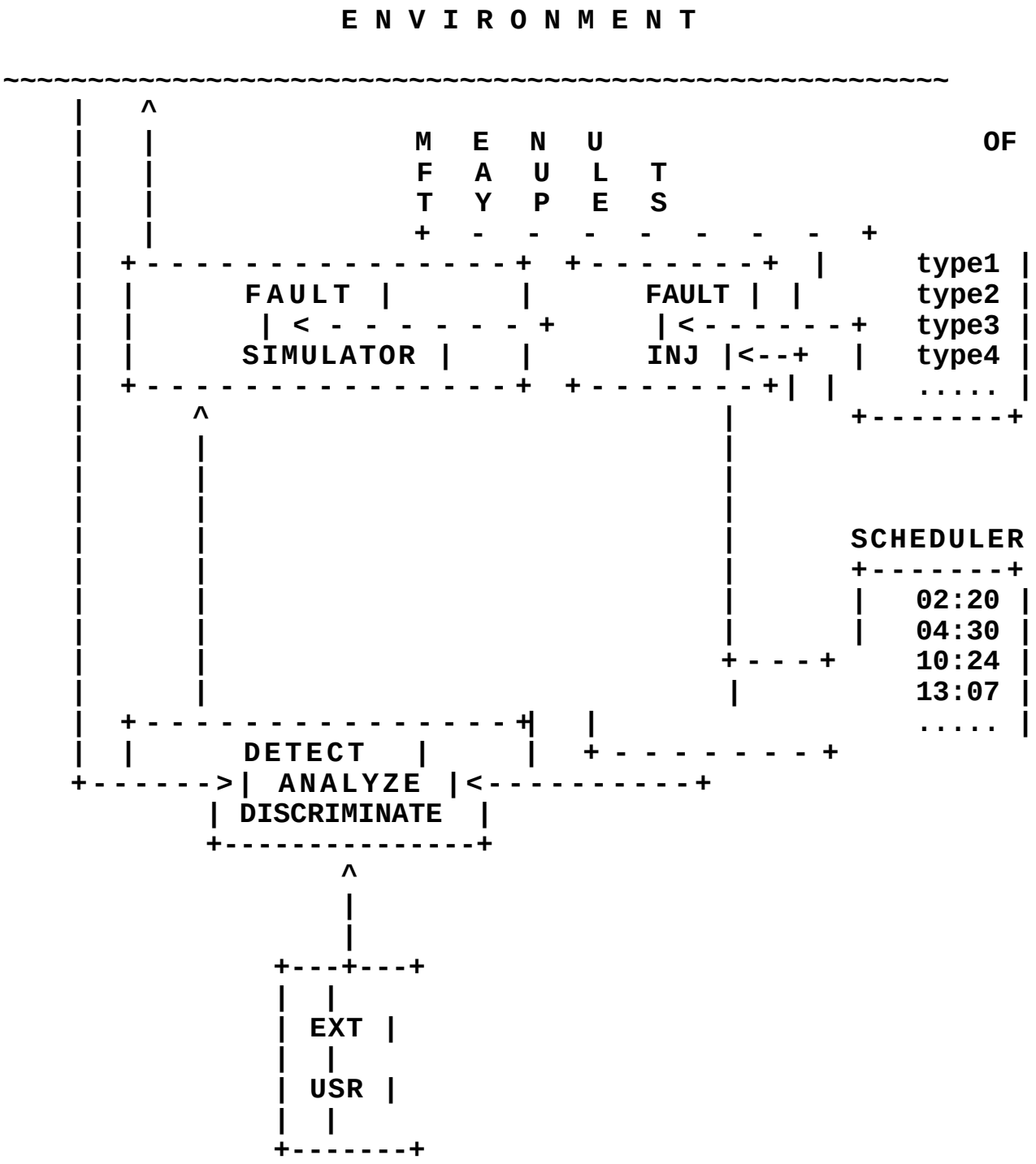
High Performance

- DEC 2100-5000, Sun-4, i486/66, HP 720/730, IBM RS6000, Alpha, Iris.

Machine performance by network

Network	System Performance Level	Clients	Servers	Total
Network 1/5FL	low	57	0	57
	high	57	1	58
				115
Network 2/3PU	low	76	2	78
	high	71	1	72
				150
Network 3/3PR	low	11	6	17
	high	14	9	23
				40
Network 4/7FL	low	18	0	18
	high	34	0	34
				52
Network 5/DOH	low	45	0	45
	high	36	1	37
				82

Blind event-injection system



Measured parameters

1- FAULT TYPE
1 = 60-Byte Pseudo-Runt Storm
2 = Network Paging
3 = Bad Frame-Check Sequence
4 = Jabber
5 = Broadcast
2- NETWORK TYPE (1, 2, 3, 4, 5)
3- INJECTION NUMBER (1-24)
4- DESTINATION ADDRESS, UNUSUAL ACTIVITY
5- DESTINATION ADDRESS, INCREASED ACTIVITY
6- DESTINATION ADDRESS, CEASED ACTIVITY
7- DESTINATION ADDRESS, SUDDEN APPEARANCE
8- SOURCE ADDRESS, UNUSUAL ACTIVITY
9- SOURCE ADDRESS, INCREASED ACTIVITY
10- SOURCE ADDRESS, CEASED ACTIVITY
11- SOURCE ADDRESS, SUDDEN APPEARANCE
12- PERCENT_UTILIZATION
13- PACKET_COUNT
14- COLLISION_COUNT
15- PACKET LENGTH < 63
16- PACKET LENGTH 64-127
17- PACKET LENGTH > 1024

Demonstrations of network measurements

- Raw collision traffic
- Traffic histograms
- Nonstationarity

Detection coverage

		Network				
		1	2	3	4	5
F	1	1.00	1.00	1.00	1.00	1.00
a	2	1.00	0.88	0.79	1.00	1.00
u	3	0.67	0.21	0.13	1.00	1.00
l	4	1.00	1.00	1.00	1.00	1.00
t	5	1.00	0.96	1.00	1.00	1.00

93.4% overall

Cross-validation probability matrix

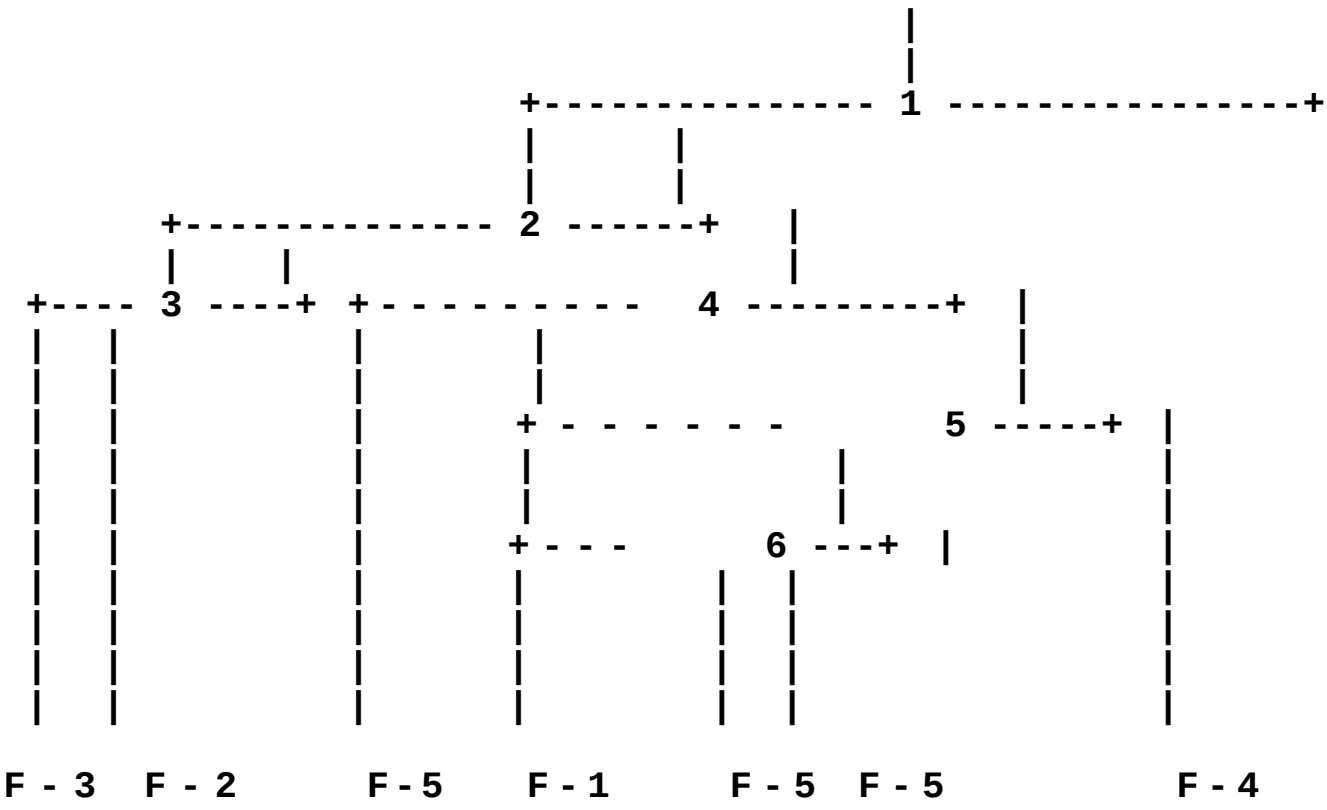
Predicted Class	True Class 1	2	3	4	5
1	0.92	0.00	0.03	0.00	0.33
2	0.01	0.89	0.07	0.01	0.00
3	0.00	0.07	0.86	0.00	0.00
4	0.00	0.01	0.01	0.98	0.00
5	0.08	0.03	0.03	0.01	0.67

86.8% overall

Automatically induced decision rule

```
Read feature vector; then start at Node 1:


Node 1: IF PL0127  .le.  5.00e-01 THEN GOTO Node 2
        ELSE fault = type 4

Node 2: IF PL0063  .le.  5.00e-01 THEN GOTO Node 3
        ELSE GOTO Node 4

Node 3: IF DESADSU .le.  5.00e-01 THEN fault = type 3
        ELSE fault = type 2

Node 4: IF DESADSU .le.  5.00e-01 THEN fault = type 5
        ELSE GOTO Node 5

Node 5: IF DESADIN .le.  1.50e+00 THEN GOTO Node 6
        ELSE fault = type 5

Node 6: IF SOUADIN .le.  1.50e+00 THEN fault = type 1
        ELSE fault = type 5
```

Classification tree



Measurement plan

1. State the goals and define the system under observation.
2. List the services and outcomes of interest.
3. Select metrics/criteria for judging performance.
4. List all of the parameters that affect performance.
5. Select factors and levels.
6. Select an evaluation technique.
7. Select a stimlus/workload.
8. Design a sequence of experiments that offers maximum information with minimum effort.
9. Analyze and interpret the data, taking into account the variability of the results, if any.
10. Present results.

Adapted from Jain 1991: Computer Systems Performance Analysis.

Synthetic anomaly-detection experiment

1. Goal: Determine accuracy of anomaly detector in a changing environment; find performance boundaries automatically.
2. Outcome: Anomaly-detection alarm
3. Metrics: Hits, misses, false alarms
4. Parameters: Model size, embedded structure, alphabet size, dataset length, event under scrutiny, threshold of surprise
5. Factors/levels: Model size (1-6), embedded structure (0-8, threshold (.80-1.0)
6. Technique: Simulation
7. Stimuli/workload: Dataset of 10,000 with .01 injected-event probability
8. Experiment: Quad-structured event to detect, varying model size, varying embedded structure, varying threshold
9. Analysis: Error probabilities
10. Presentation: ROC curves

Advantages of using synthetic data

- Controlled experimentation.
- Time compression.
- Sensitivity analysis.
- Undisturbed real system.

Simulation pitfalls

- Inappropriate level of detail
- Unverified models
- Invalid models
- Improperly handled initial conditions
- Foreshortened simulations
- Poor random-number generators
- Improper selection of seeds

Experiment details

- Test of detection accuracy in a changing environment; determine whether or not real-time tuning is necessary.
- Alphabet size = 6
- Dataset length = 10,000
- Environment = stationary, uniform random
- Anomaly = quad structure
- Injection scheme - replace 1 of original sequence
- Seeds - same for training and test sets
- Ground truth generated automatically
- Data sets can be replicated easily for sharing - Cinnamon
- Error analysis - TBD
- Real-time lag / Response time = immediate
- System resources - N/A

About the Cinnamon synthesizer

The Cinnamon data synthesizer ...

- Provides a flexible palette of random number generation techniques.
- Provides an interface tailored to facilitate stochastic process modeling.
- Can also be used for less complex applications.

Basic innovations:

1. Uses Marsaglia's lagged Fibonnaci product algorithm, the best current practice for generating uniform random numbers.
2. The interface design makes it easy to introduce drift, intrusions, perturbations, and complex serial dependence.
3. It is easy to generate AR(p), MA(q), and ARMA(p,q) time series models. These models handle special kinds of process correlation.
4. Offers the only Markov generator available in any package of which we know.

Using Cinnamon for system evaluation

Regarding any process that generates time-series data:

Provide for ...

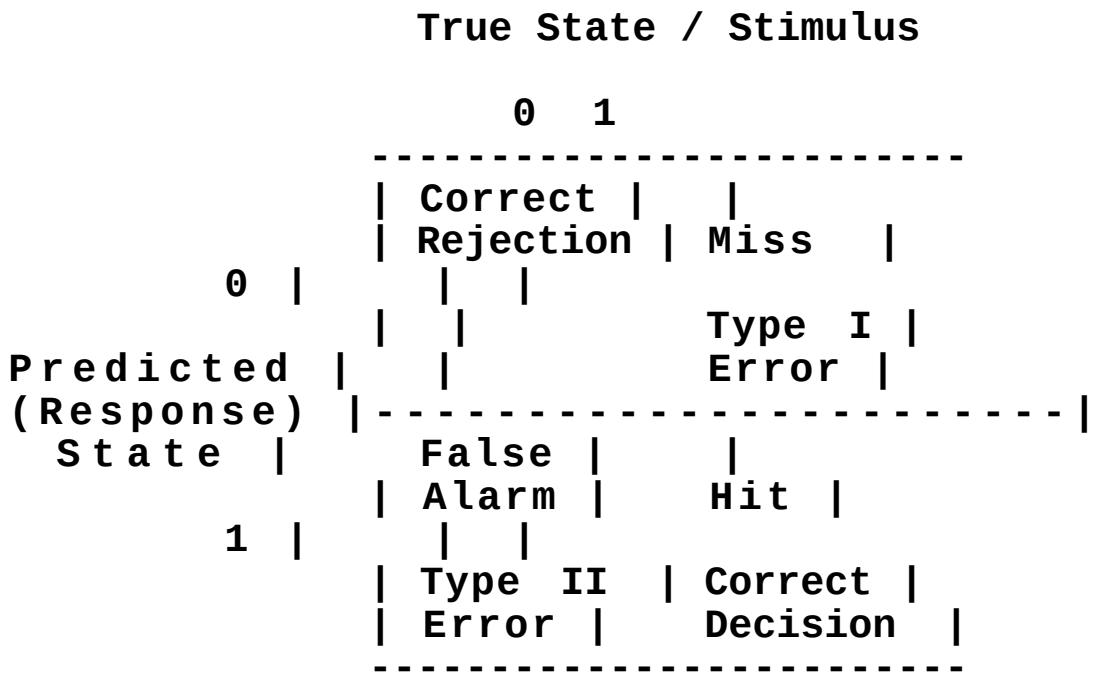
- Different types of events.
- Range of discriminability among events.
- Range of drift in environment.
- Range of noise in environment.
- Range of data variability from epoch to epoch.
- Variable-selection capability.
- Latent-variable recognition.
- Range of injected perturbations.

Cinnamon portable specification file

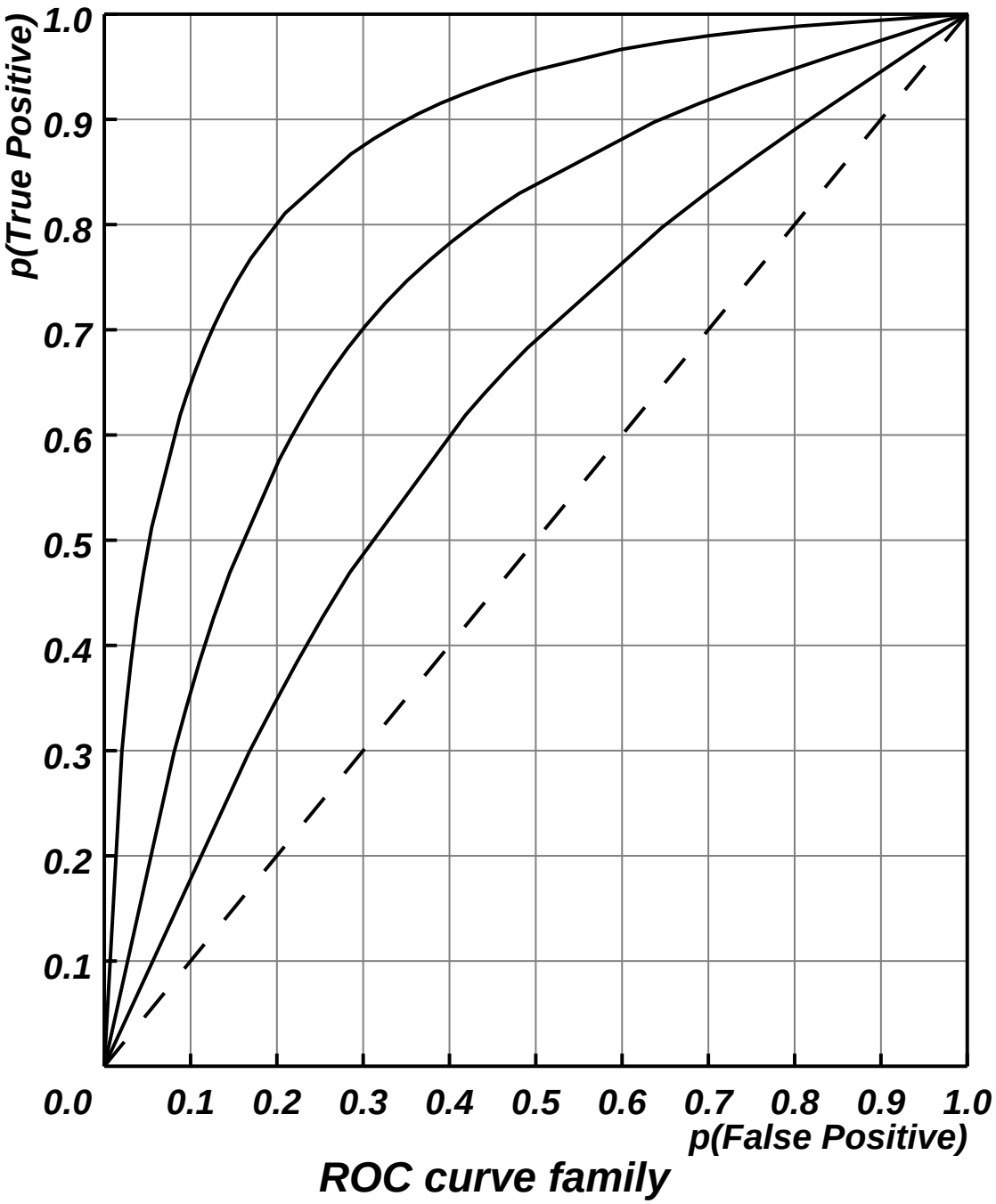
```
; Configuration Parameters
SEED=1234
DISTRIBUTION=MULTINOMIAL
VECTORS=1
PRECISION=0
LABELS=NO
OUTPUT_FILE_NAME=/usr/cinnamon/archive/980906171640/synth#.dat
;
; Continual Data Generation Parameters
;
FREQUENCY=-1
;
; Perturbation Parameters
;
PERTURBATIONS=0
DATAPOSITION=1
QUANTITY=10000
;
; Drift Parameters
;
DRIFT=0
USERVALUE_1=0
USERVALUE_2=0
USERVALUE_3=0
DRIFT_ONSET_INDEX=0
DRIFT_OFFSET_INDEX=0
INDEX_INTERVAL=1.0
INDEX_START=1.0
;
; Serial Correlation Parameters
;
SERIAL_CO=NO
;
; Distribution-Specific Parameters: MULTINOMIAL
;
MULTINOMIAL_NUM_BINS=6
MULTINOMIAL_PROB_MATRIX=1,1|.1666
MULTINOMIAL_PROB_MATRIX=2,1|.1666
MULTINOMIAL_PROB_MATRIX=3,1|.1666
MULTINOMIAL_PROB_MATRIX=4,1|.1666
MULTINOMIAL_PROB_MATRIX=5,1|.1666
MULTINOMIAL_PROB_MATRIX=6,1|.1666
MULTINOMIAL_CLASS_MATRIX=1,1|A
MULTINOMIAL_CLASS_MATRIX=2,1|B
MULTINOMIAL_CLASS_MATRIX=3,1|C
MULTINOMIAL_CLASS_MATRIX=4,1|D
MULTINOMIAL_CLASS_MATRIX=5,1|E
MULTINOMIAL_CLASS_MATRIX=6,1|F
```

Event-detection (stimulus-response) outcomes

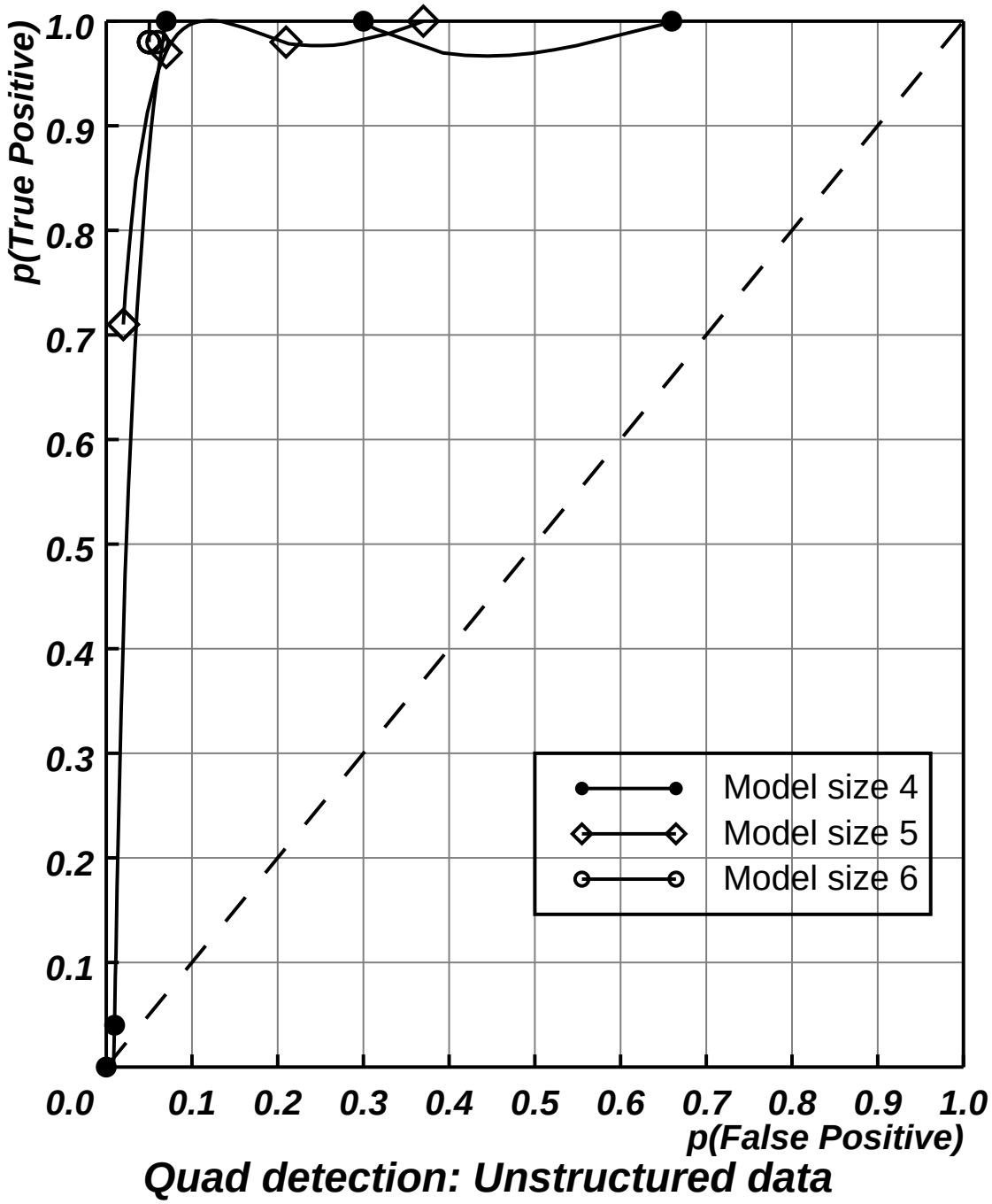
- Hits
- Misses
- False alarms
- Correct rejections



Relative operating characteristic (ROC) curves

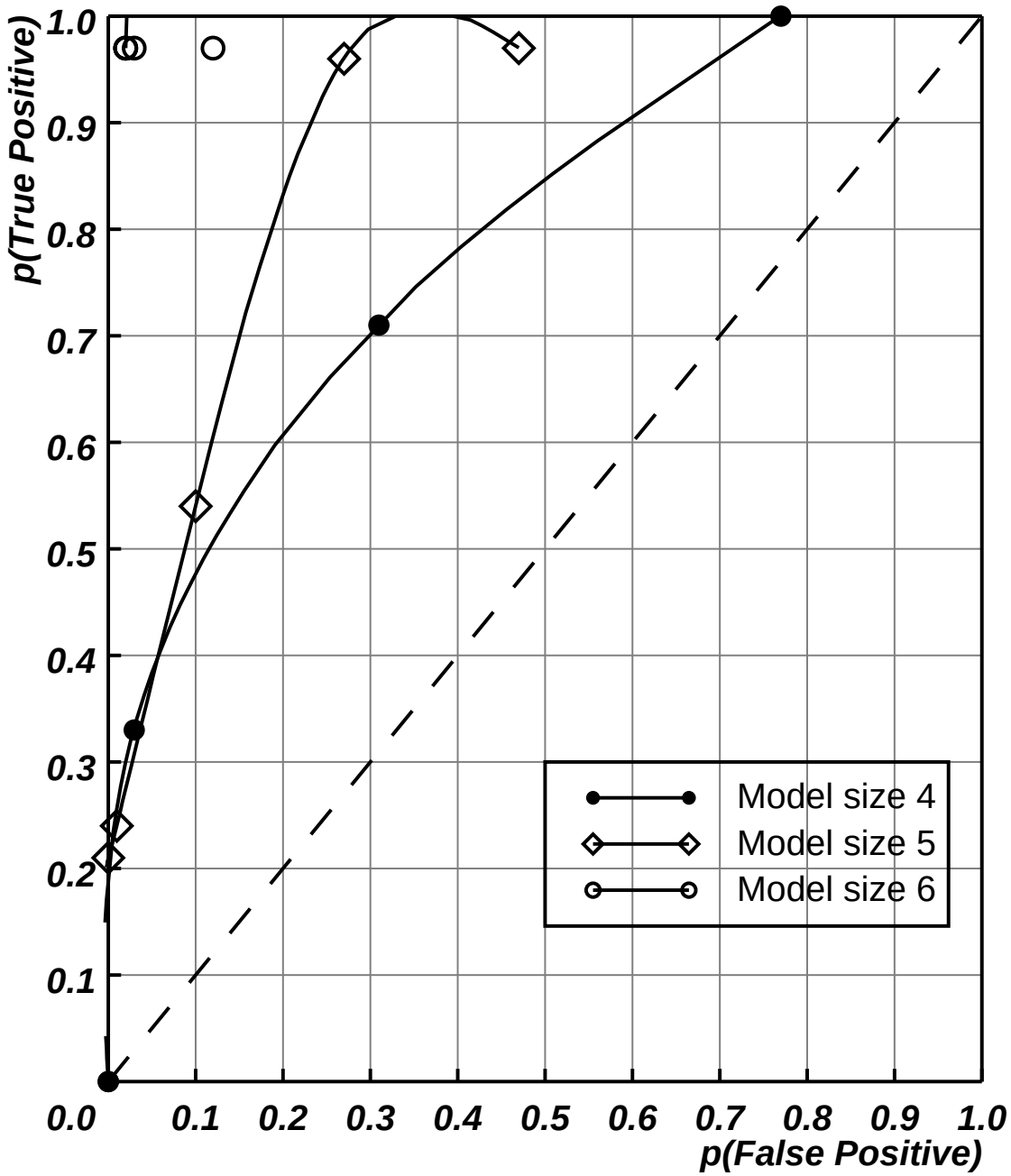


Detection ROC - No structure



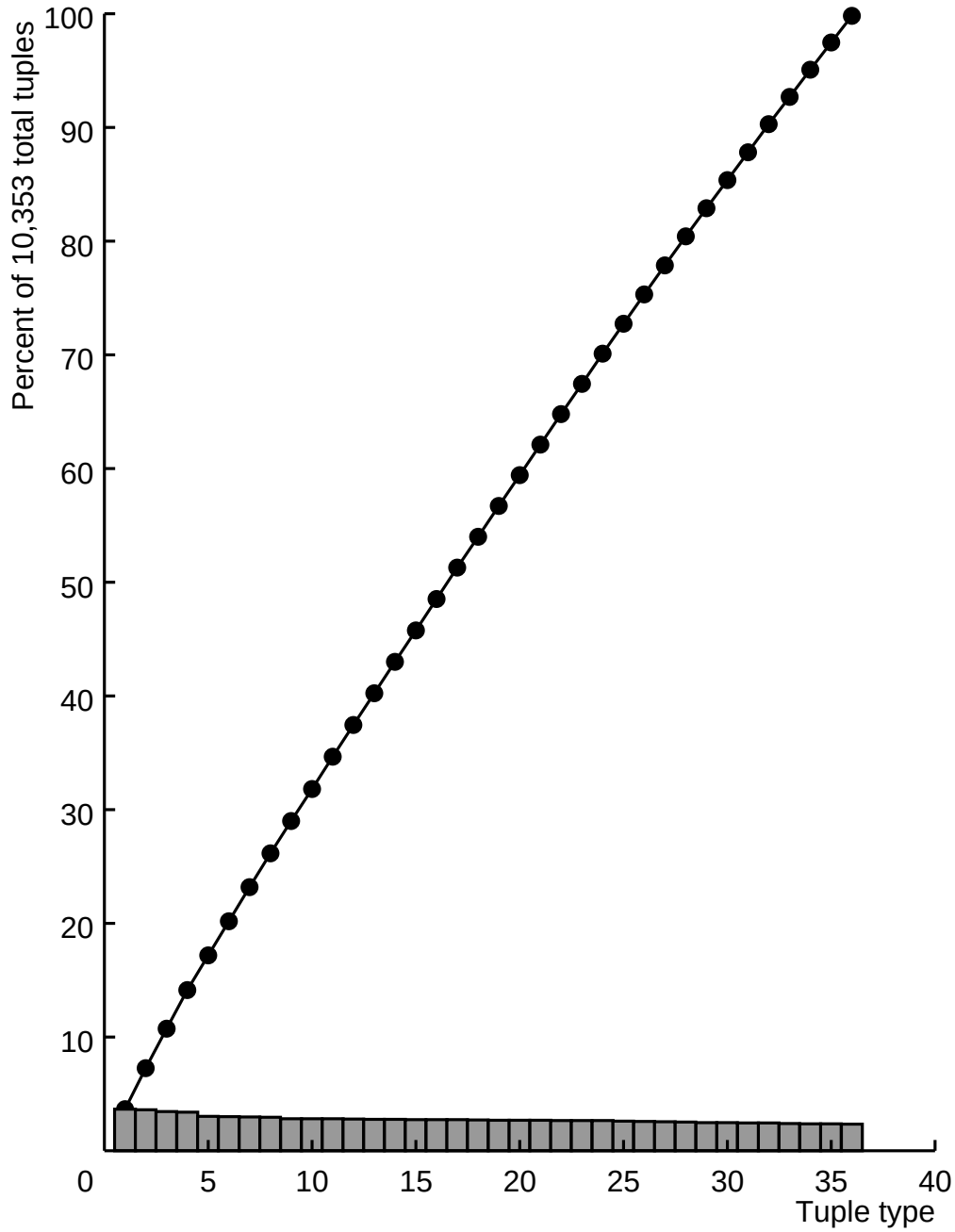
Quad detection: Unstructured data

Detection ROC - Diad structure



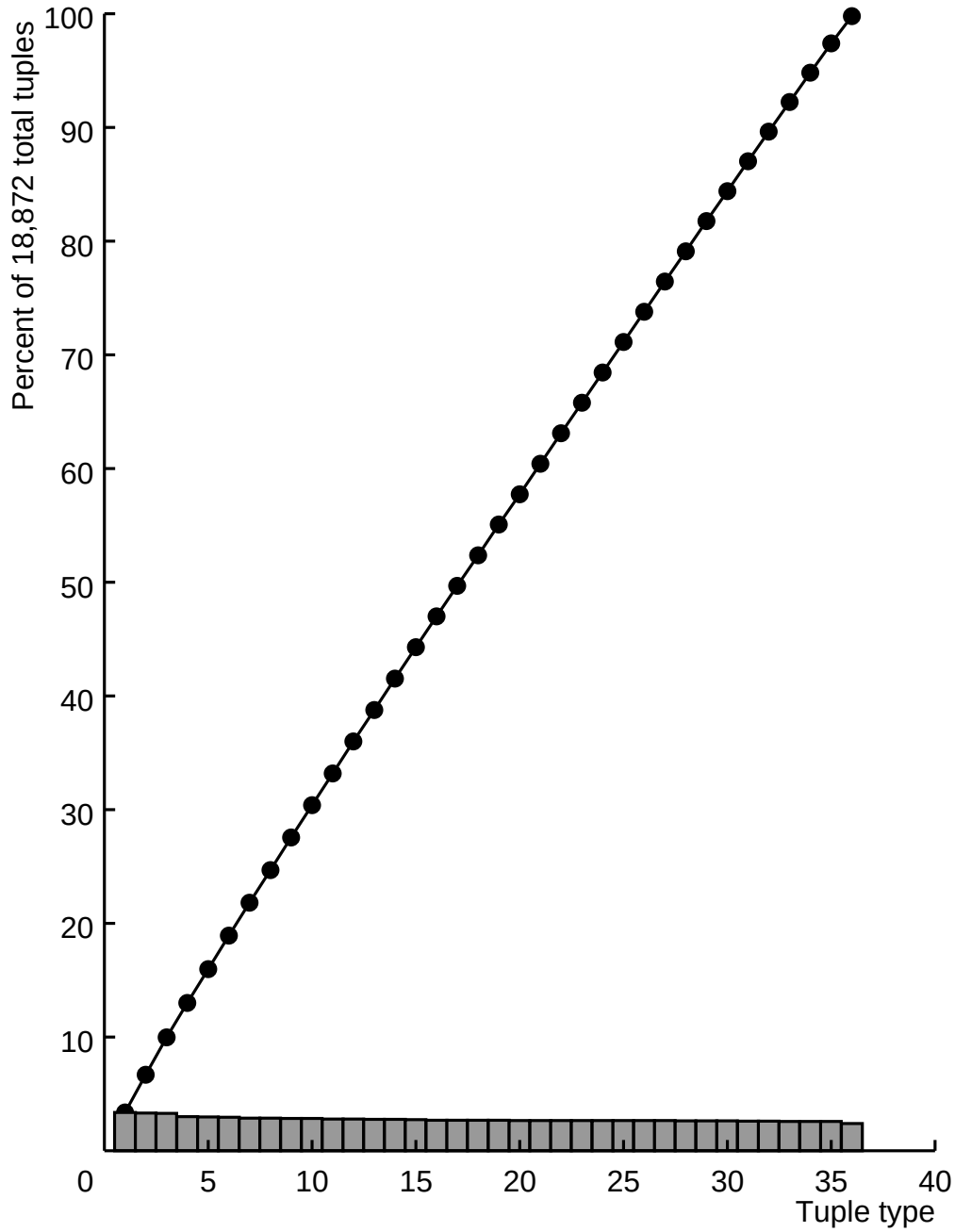
Quad detection: Diad-structured data

Diad coverage: unstructured data



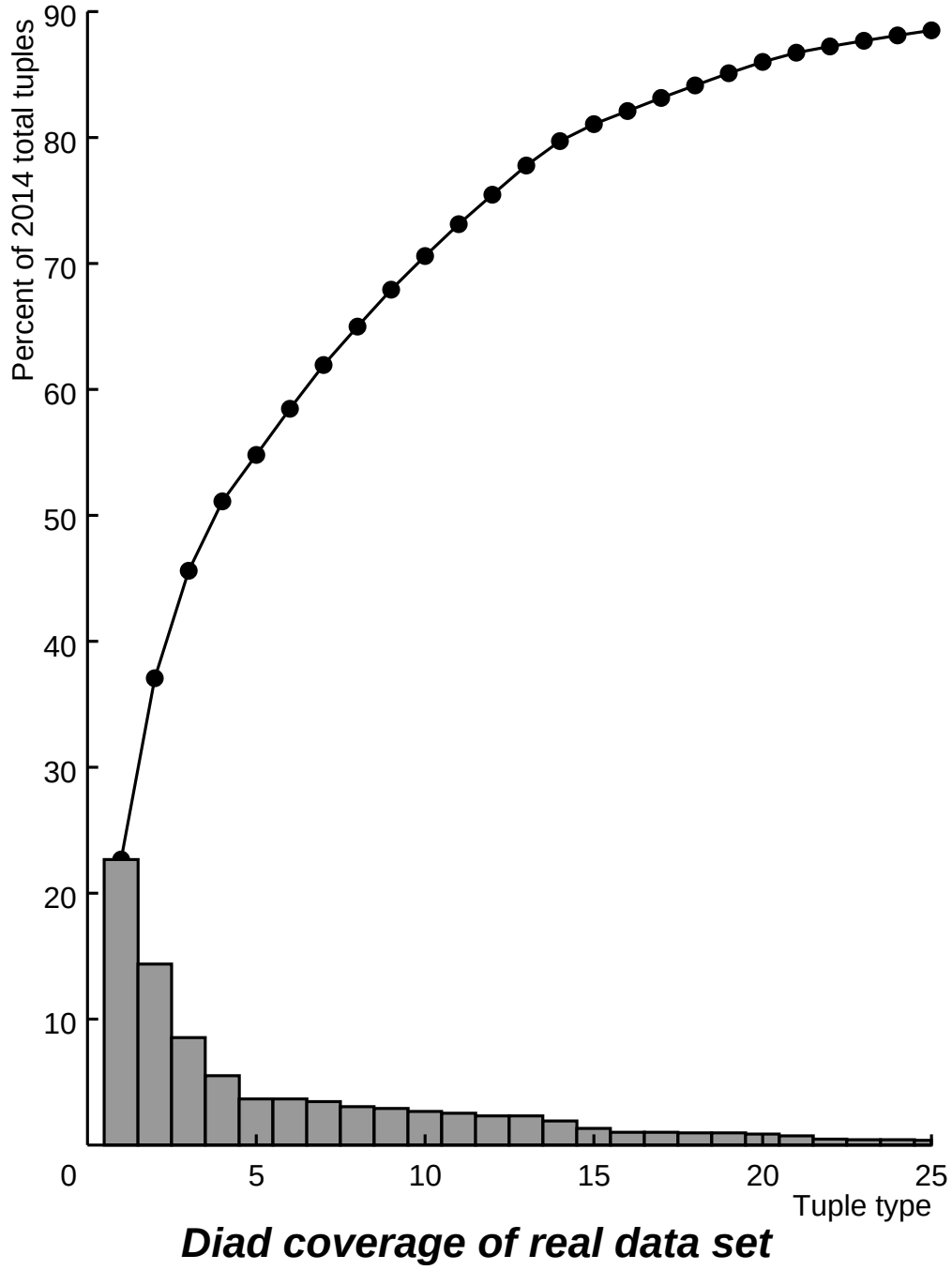
Diad coverage of unstructured data set

Diad coverage: diad-structured data



Diad coverage of diad-structured data set

Diad coverage: real data



A decade of progress in intrusion detection

Since Dorothy Denning's 1987 paper ...

- Papers published
- Systems implemented (45)
- Trend toward commercial deployment
- Conceptual or paradigmatic shifts
- Empirical measures

What facilitates progress?

Incremental advancement vs. paradigm shift ...

- Incremental advancement
  - Quantification
  - Counting and measurement
  - Careful definition
  - Scientific method -
    - Planck’s discovery of the law of blackbody radiation
    - Adams & Leverrier’s discovery of the planet Neptune
    - Ohm’s law of electrical resistance
    - Kepler’s laws of planetary motion
- Paradigm shift (serendipitous & revolutionary)
  - Kekule’s dream of the benzene ring
  - Fleming’s discovery of penicillin
  - Darwin’s theory of evolution
  - Einstein’s theory of relativity

In the absence of paradigm shift, measurement incrementally paves the path of progress.

Conclusion

- Measurement isn’t always exciting, and it can be tedious.
- We progress from art to craft to engineering to science.
- Without measurement, instrusion detection remains, at best, a craft; more likely an art.
- An experiment well done, and convincingly measured, conveys more information than anecdotal stories do.
- Measurement permits us to mark progress and to know what the open problems are.
- Quantitative data/results allow our successors to see farther and go faster by standing on our shoulders.