Network Metrics and Measurements in the Era of the Digital Economies

Pavel Loskot
p.loskot@swan.ac.uk

Swansea University
Prifysgol Abertawe

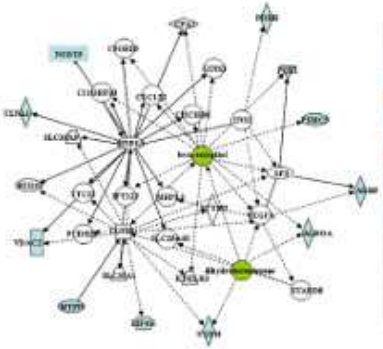
Background

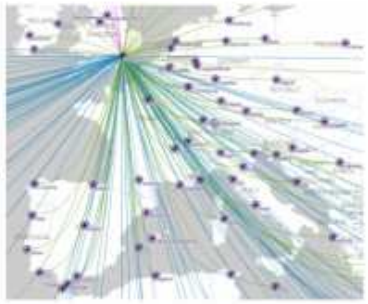
I am

- Engineer with 20 yrs experience, mostly in signal processing and telecommunications
- Senior Lecturer at Swansea University, in Wales, United Kingdom

Some types of projects I am/was involved in

- telecommunication networks: from signals to protocols
- social networks: broadband network subscribers behavior and forecasting
- biological networks: whole-cell simulations
- air-transport networks: load optimization
- utility grids: solving energy crisis in Nepal

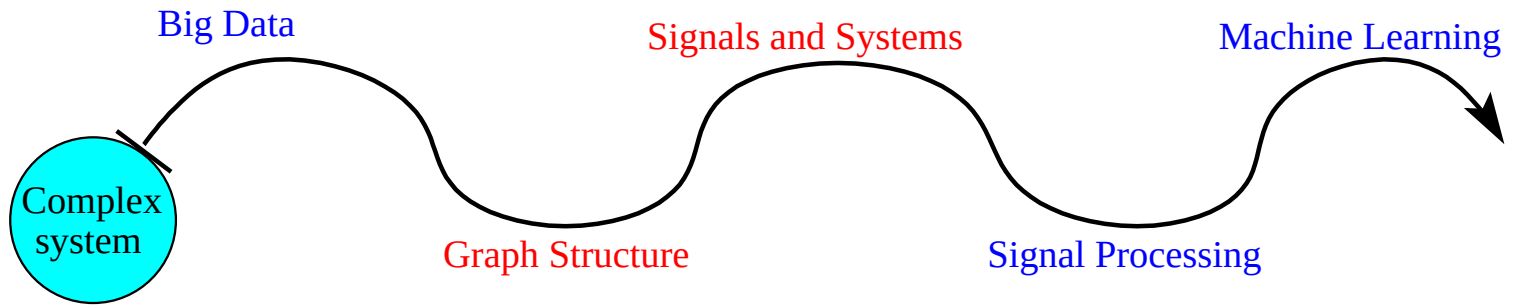




COURSE OUTLINE

Metrics and measurements for heterogeneous telecommunication networks:

1. defining, selecting standardizing and categorizing metrics
2. key performance indicators, service level agreements, metrics frameworks
3. measurement procedures, conditions, assumptions and strategies
4. perceptions of stakeholders
5. system models and abstractions, constraints, optimality
6. system sensing, projections, learning versus inference, sensitivity analysis
7. measuring the digitalized systems within the Digital Economy

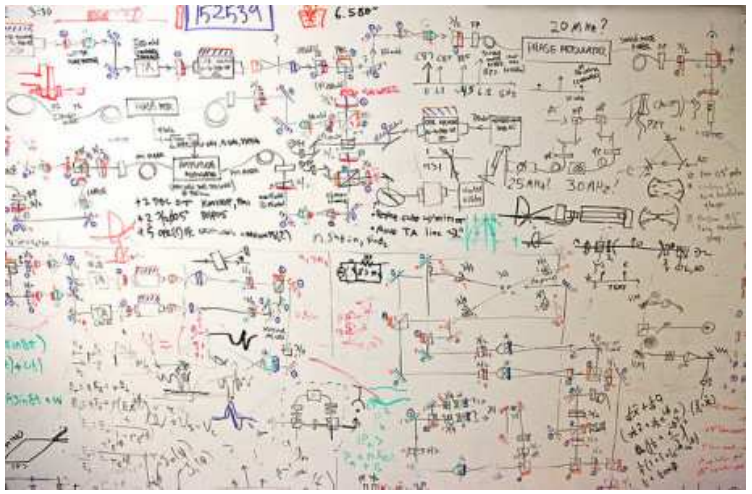


Background knowledge

Complex systems [1]

Key characteristics

- far beyond complicated (size irrelevant)
- extremely difficult to understand, analyze or just describe
  → car is complicated
  → computer is both complicated & complex
  → brain is very complex
- easily unpredictable if non-linear interactions
  → self-organization, emergence, chaos
- complexity increasing with time → evolution
- equilibrium vs. steady-state



Tools and methods

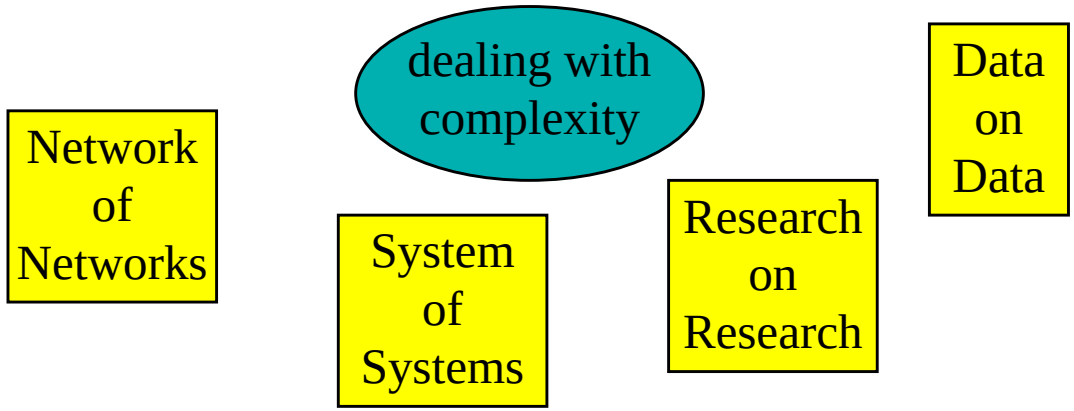
- usual methods not efficient, or not work at all:
  → trial & error, divide & conquer
- scales: micro → meso → macro

Wicked problems

- every such problem is unique
- can never be completely solved, solutions only better/worse
- examples: terrorism, healthcare, energy, global warming

[1] SH Kaisler, G Madey, Complex Adaptive Systems: Emergence and Self-Organization, U. Notre Dame, 2009.

COMPLEX SYSTEMS (2)



Strategies

- iterative methods
- hierarchy
- dimensionality reduction
- model simplification (e.g. approximation)
- transformations
- new/alternative representations
- standardization
- encapsulation
- ...

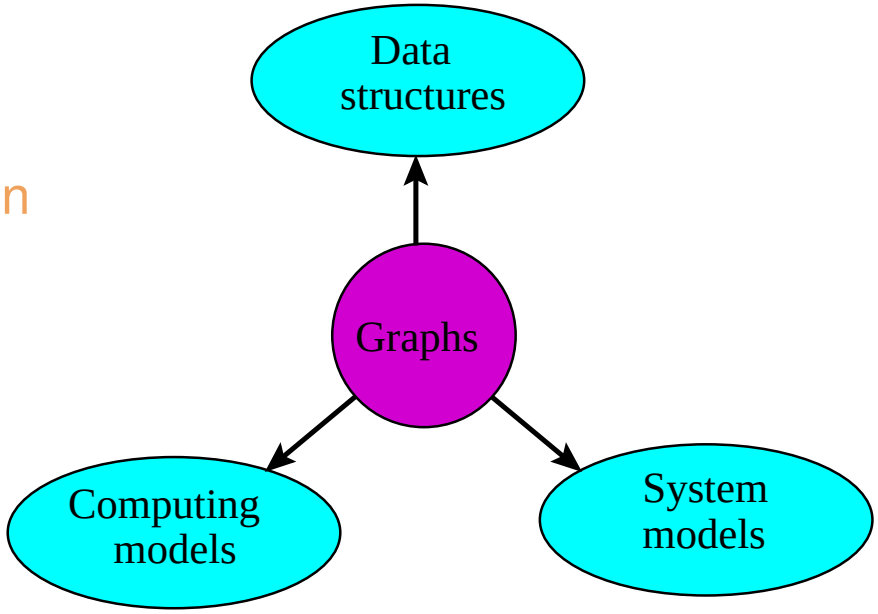
UNDERSTANDING GRAPHS

Graphs as data structures

- discrete mathematical structures (tensors, lists, trees, ...)

Graphs as a computing model → function

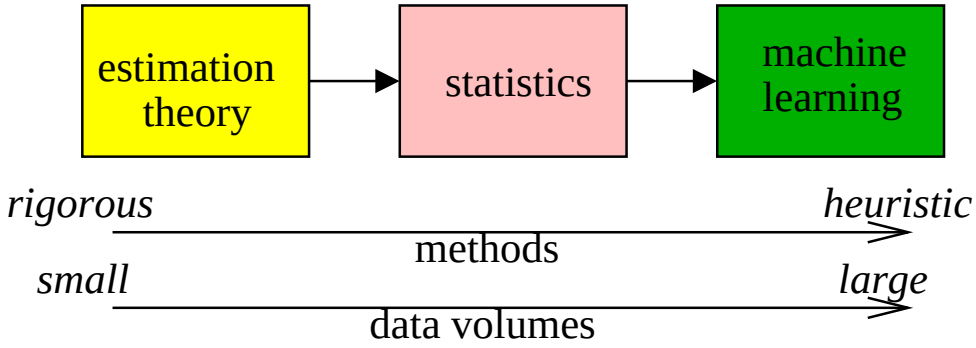
- Gaussian networks
- Bayesian networks
- Markov random fields
- hidden Markov models
- finite state machines



Graphs as a system model → structure

- telecommunication networks and WWW
- utility grids
- transportation networks

Tools and toolkits



Signals and Systems

- concerned about mathematical models for signals and for systems

Signal processing

- functions of numbers
- numbers arranged into regular structures (scalars, vectors, matrices)
- convenient algebraic operations for these regular data structures
- outcome is an algorithm

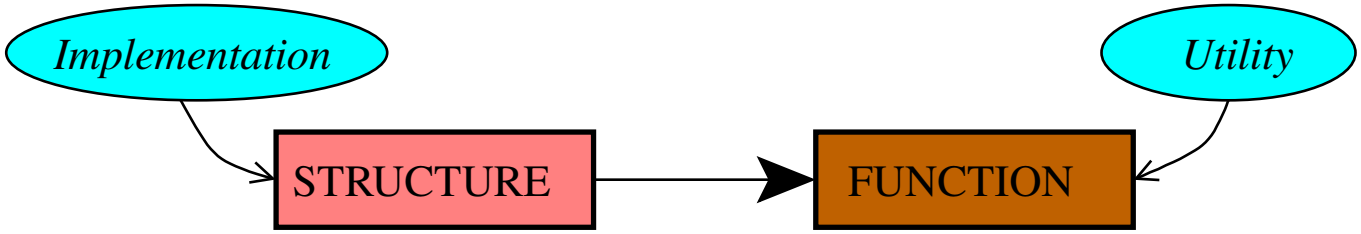
Machine Learning

- arbitrary data structures, a lot of heuristic approaches
- the aim is discovery of patterns, relationships and knowledge

Computer programming

- many data structures and associated algorithms for basic operations

Structure vs. function



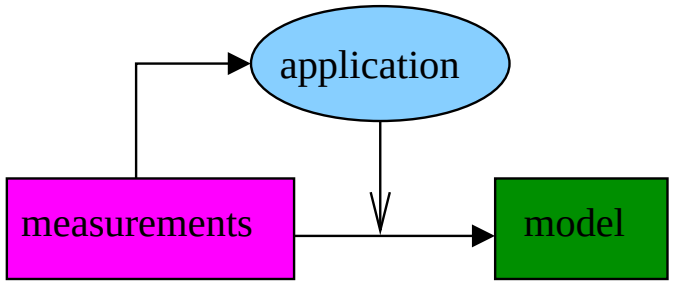
Analysis strategy

- known tuples *(structure,function)* to train ML classifier (e.g. Deep Learning)
- common in biochemistry to predict protein function
- can replace Deep Packet Inspection with cheaper and faster ML classifiers

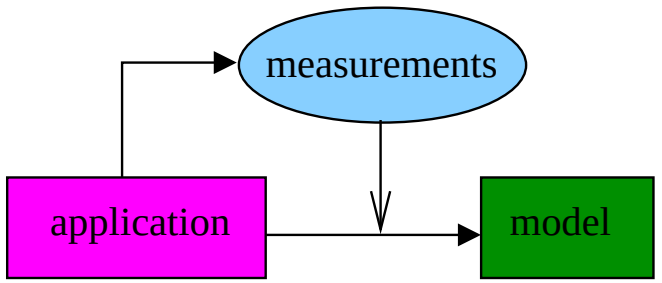
Synthesis strategy

- little explored Engineering territory

Reverse (data-driven) vs forward (application-driven) modeling

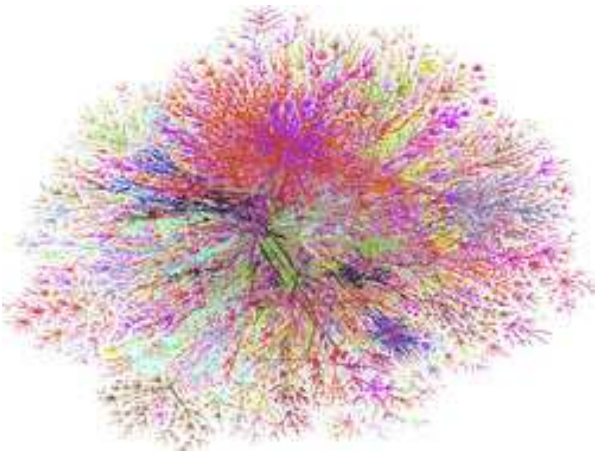


available measurements constrain possible applications



application determines required measurements

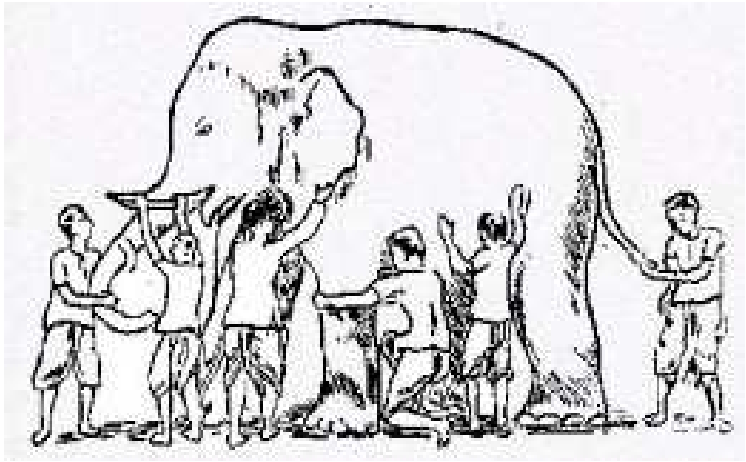
MODELING



1. Micro-level: organized simplicity (machines)
2. Meso-level: organized complexity (systems)
3. Macro-level: unorganized complexity (aggregation)

Models

- *"all models are wrong, but some of them are useful"* [G. Box, 1976]
- *"With four parameters I can fit an elephant, and with five I can make him wiggle his trunk."* [J. von Neumann]
- a complex system is often associated with (infinitely) many models
- challenges
  - not knowing enough (e.g. inner interactions)
  - dynamics (emergence, self-organization)



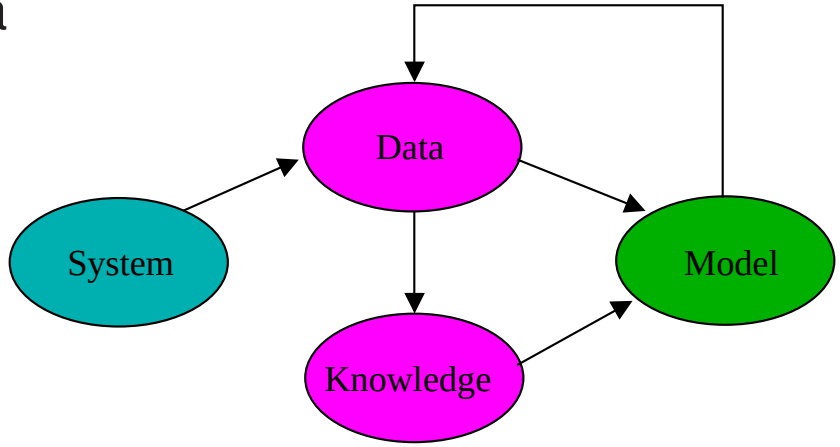
Optimum modeling

Objectives

- aim for systematic and consistent approach
  - e.g. carefully consider what is known and what is not
- optimality → is it even possible? in what sense? what constraints?
  - if two models, how to choose the better one?
- enable automation of modeling

Optimum modeling strategy

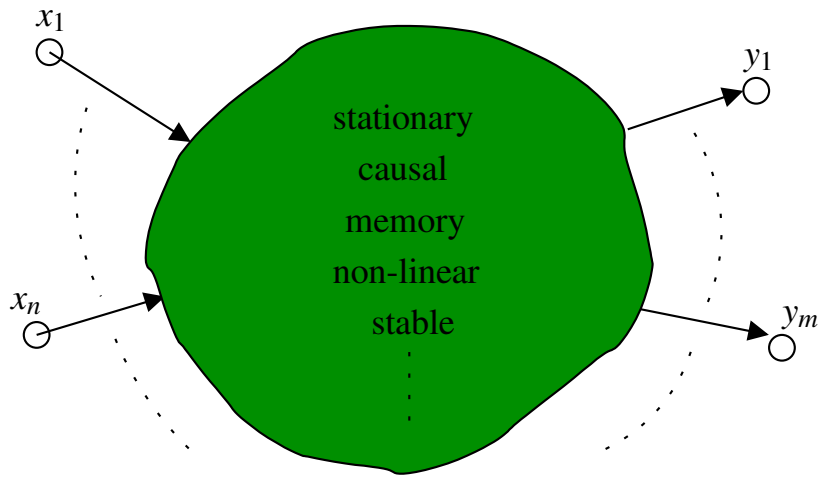
1. no knowledge of system, but plenty of data (Big Data)
   → reverse (backward) modeling
2. some knowledge of system, but no data
   → forward modeling
3. some knowledge of system with some data
   → typical situation



Data collection consistent → more data, better model

ASSUMPTIONS

Model



definition:

input-output dependencies

x_n inputs (implicit or explicit) are (mostly) independent; they configure the system

y_m outputs (observations, measurements)

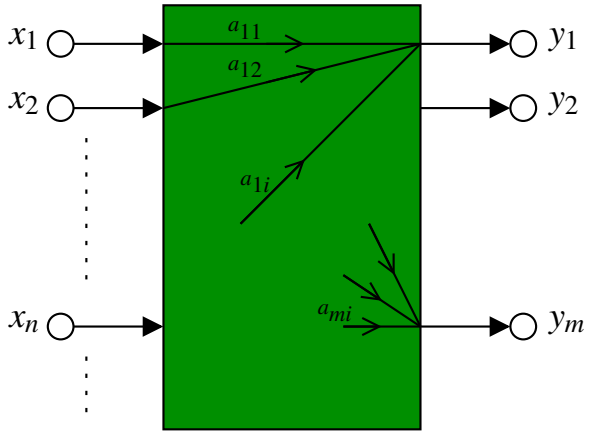
both: are assumed to be a random variable or a random process; there are too many of them (countably infinite), but only some of them are known or used

Model limitations

- many, e.g. no dynamics (of structure or function)

Assumed model

Bipartite directed graph



Dependency matrix

	x_1	x_2	$\cdots$	x_n	$\cdots$
y_1	a_{11}	a_{12}		a_{1n}	
y_2	a_{21}	a_{22}		a_{2n}	
$\vdots$			$\ddots$		
y_m	a_{m1}	a_{m2}		a_{mn}	
$\vdots$					

$a_{ij} \in \{0,1\}$

System (output, input-output, input-combining) functions

$$y_j = f_j(x_1, x_2, \ldots, x_i, \ldots), \quad j = 1, 2, \ldots$$

- only some f_j are of interest (given by specific application)
- some inputs affect output y_j significantly, others negligibly or not at all ($a_{ij} = 0$)
- many inputs often neglected due to our lack of knowledge about the system
- inputs x_i may change dependencies a_{ij} (ignored for now)

More on inputs

Classification of inputs x_i

- x_i known to affect y_j, and may or may not be ignored $\rightarrow$ nuisance inputs
- existence of x_i is unknown $\rightarrow$ possibly large modeling error
- implicit inputs are similar to outputs as they are also observed or measured i.e. inferred from the system (e.g. an input to set the system temperature)
- some inputs may constrain other inputs (e.g. bound by thermodynamic laws)
- inputs x_i are otherwise independent, or have controlled dependency; e.g. defined cross-covariance matrix

$$\mathrm{E}\Big[(\mathbf{x}-\bar{\mathbf{x}})(\mathbf{x}-\bar{\mathbf{x}})^T\Big]$$

  where $\mathrm{E}[\cdot]$ is expectation, and $\bar{\mathbf{x}} = \mathrm{E}[\mathbf{x}]$
- an input x_i is considered and treated as
  1. a (random) constant over time
  2. relatively slowly varying over time
  3. relatively fast varying over time

observations y_j often averaged over fast varying x_i, so $y_j = \mathrm{E}_{x_i}\Big[f_j(x_1,\ldots,x_i,\ldots)\Big]$

More on system functions

Estimated system functions

$$y_j = \hat{f}_j(x_1, x_2, \ldots, x_n) \;\approx\; f_j(x_1, x_2, \ldots, x_n, \ldots)$$

- dependency on input factors $(x_1, x_2, \ldots, x_n)$ is explicitly considered; other factors are intentionally or unintentionally ignored
- all plausible factors $(x_1, x_2, \ldots, x_n)$ form a n-dimensional input space
- $\hat{f}_j$ is (hopefully) a good approximation of true f_j

Methods to remove x_i from $f_j(\ldots)$

(output post-processing or modeling strategy)

1. assume $y_j = \mathrm{E}_{x_i}\Big[f_j(\ldots)\Big] \rightarrow$ averaging (or other statistics)
2. assume $y_j = \int f_j(\ldots)\,\mathrm{d}x_i \rightarrow$ accumulation
3. assume $y_j = \min_{x_i} f_j(\ldots) \rightarrow$ optimization
4. set x_i to specific value $\rightarrow$ conditioning or parametrization
5. ignore x_i and assume $\hat{f}_j$ instead of $f_j \rightarrow$ approximation

Data processing

Inputs x_i and outputs y_j

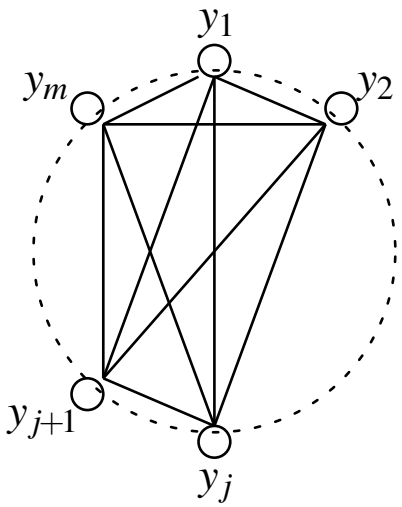
- samples of random variables or random processes
- OR, statistics inferred from these samples

Typical data operations

- space-time indexing (labeling) $\rightarrow$ space-time alignment $\rightarrow$ causality
- scaling, outliers, missing values, inferences (noise suppression)
- give rise to new nodes (and corresponding dependencies) in our system model

$$x_i \mapsto x_i' \qquad y_j \mapsto y_j'$$

Pairwise output dependencies



- if $y_j = f_j(x_1, \ldots, x_n)$ and $y_{j+1} = g(y_j)$ then

$$y_{j+1} = g(f_j(x_1, \ldots, x_n)) \quad \text{(compounding)}$$

$$(x_1, \ldots, x_n) \mapsto y_j \mapsto y_{j+1}$$

- more generally, combined outputs

$$f_1 = g(f_2, \ldots, f_m)$$

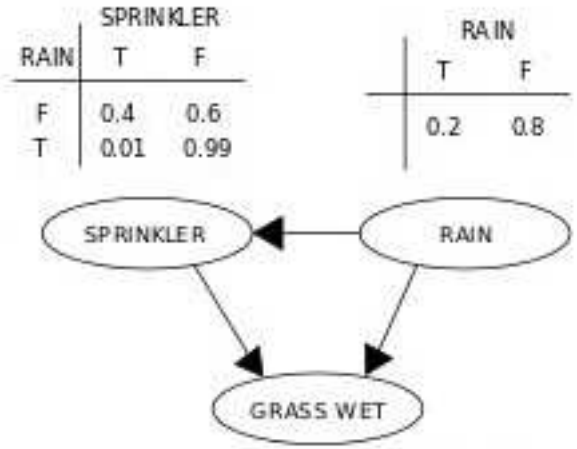
OVERALL MODEL

Multiple-root directed tree

- inputs x_i are the leaf nodes
- observation nodes y_j can be further combined
- edges associated with system functions $y_j = f_j(\ldots) \rightarrow$ represent flows

Other similar (graphical) models

- inner states represented as random variables
- many algorithms available for making statistical inferences
- for example
  - (hidden) Markov models and random fields
  - Bayesian networks



SPRINKLER	RAIN	GRASS WET T	GRASS WET F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Model applications

- due to stochastic inputs and outputs, we can
  1. test hypotheses
  2. quantify dependencies

Model applications

Modeling

- consider specific inputs $x_1, x_2, \ldots, x_n$ and a specific output y
- then find system function f such that

$$y = f(x_1, x_2, \ldots, x_n)$$

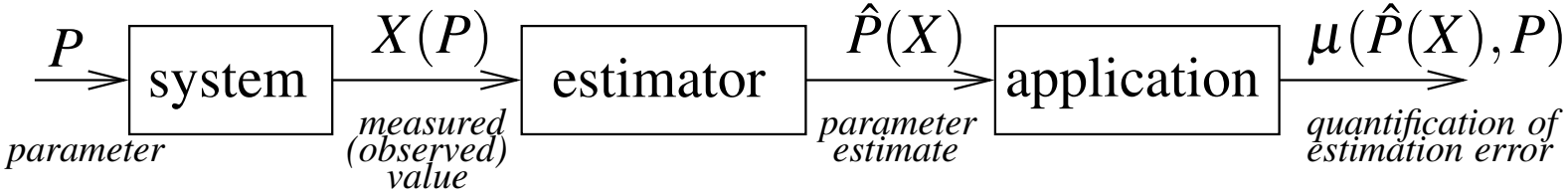
- f is constrained by other known or inferred (from data) system functions f_j
- how - to be discussed

Hypothesis testing

- given a set of specific system functions $f_1, f_2, \ldots, f_j$ with specific but possibly different inputs x_i, identify the function that best describes observed data y
- how - only briefly discussed here
- OR, given a specific system function $y = f(x_1, x_2, \ldots, x_n)$, identify a subset of inputs x_i that influence the output y the most $\rightarrow$ sensitivity analysis
- OR, given system function f and observed data $y = f(\ldots, x_i, \ldots)$, find the most likely value of unobserved input $x_i \rightarrow$ inference

DATA/SIGNAL PROCESSING TASKS

Inference



- system function $X(P)$ is known, sometimes also a priori statistic of P is known
- task: given model $X(P)$ and output observations, find the estimate $\hat{P}$ of P to minimize the risk $\mathrm{E}\left[\mu(\hat{P}(X),P)\right] \rightarrow$ optimum

Adaptive inference

- adapt to changes in statistics of P and/or X, but system model (function) $X(P)$ assumed unchanged $\rightarrow$ no learning
- although changes in statistics may lead to different optimum estimators $\hat{P}(X)$

Learning

- follow changes in system model (function) $X(P) \rightarrow$ more than adaptation
- given data y and inputs x_i, learn f, so that $y \approx f(\ldots,x_i,\ldots)$ (in some sense)

Data/signal modeling

Signal generators

- estimate statistics of the observed data/signal y; then find a system function to generate data/signal with these statistics
- N.B., different random data/signals may have the same statistics; which of these data/signals to consider is application dependent

Example scenarios:

- uncorrelated data $\mathbf{y}$:
  given $\text{cov}[\mathbf{y}] = \mathbf{A}\mathbf{A}^T$ and $\text{E}[\mathbf{y}] = \mathbf{0}$, let $y = \mathbf{A}\mathbf{u}$ where $\text{cov}[\mathbf{u}] = \mathbf{I}$ and $\text{E}[\mathbf{u}] = \mathbf{0}$
- given CDF $F_y(y)$ of y and CDF $F_x(x)$ of x:
  assume transformation $y = f(x)$ where $f(x) = F_y^{-1}(F_x(x))$
- given autocorrelation of y:
  ARMA model with Gaussian signals/data can be assumed
- given CDF as well as autocorrelation of y:
  uncorrelated Gaussian signal $\rightarrow$ linear filter $\rightarrow$ memoryless non-linearity $\rightarrow$ y

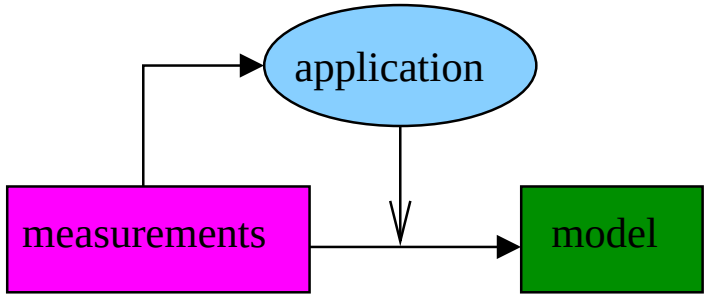
Regression

- best fit of the observed input-output data x-y to a given function f; some arguments of f are input data, other arguments are parameters of the fitting
- model linearization: find function g, so that $g(f)$ is a linear function

Reverse (data-driven) modeling

Given

- modeling application, i.e., specific inputs $x_1, \ldots, x_n$ and the associated output y
- and measurements $\{x_i\} \mapsto y_j,\ j = 1, 2, \ldots$



available measurements constrain possible applications

Task

- find system function $y = f(x_1, \ldots, x_n)$

Example

- desired (target) model: $y = f(x_1, x_2, x_3)$
- available measurements: $y_1 = f_1(x_1, x_2)$, $y_2 = f_2(x_1, x_3, x_4)$, $y_3 = f_3(x_3, x_4, x_5)$
- obtain additional system functions: $y_2' = f_2'(x_1, x_3)$ and $y_3' = f_3'(x_3)$
- hence, many models $y = f(x_1, x_2, x_3)$ satisfy the system function constraints f_1, f_2' and f_3'; e.g., let $f \equiv w_1 f_1 + w_2 f_2' + w_3 f_3'$
- additional constraints: f is unbiased, or f produces the smallest variance of y
- we can also test input independence (factorization hypothesis): $f \approx g_1(x_1, x_2) g_2(x_3)$, $f \approx g_1(x_1, x_3) g_2(x_2)$, or $f \approx g_1(x_2, x_3) g_2(x_1)$

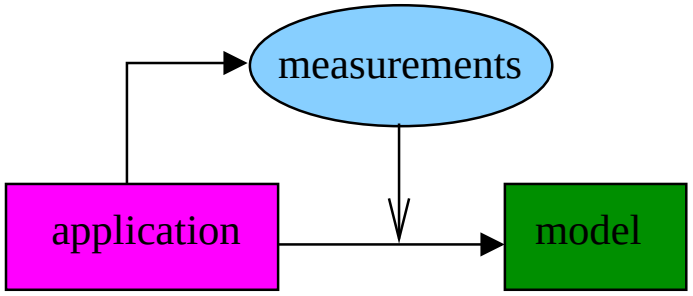
Forward (application-driven) modeling

Given

- modeling application, i.e., specific inputs $x_1, \ldots, x_n$ and the associated output y

Task

- define measurements $\{x_i\} \mapsto y_j,\ j = 1, 2, \ldots,$ to efficiently obtain system function $y = f(x_1, \ldots, x_n)$
- thus, efficiently sample n-dimensional input space $\rightarrow$ minimum measurements



application determines required measurements

Example

- desired model: $y = f(x_1, x_2, x_3)$ where x_1 is fast varying, x_2 is slow varying and x_3 dynamics are unknown $\rightarrow$ best to assume it is slow varying
- one strategy: average x_1 out of f to reduce input dimensionality from 3 to 2
- another strategy: test different factorizations of f
- other strategies can be adopted from sensitivity analysis

Sensitivity analysis

Tasks

- identify important factors, calibrate models, design experiments, formulate applications, pre-screening of factors prior to full analysis etc.

Challenges

- generating high-dimensional inputs may be expensive
- generating the output may be expensive

Methods

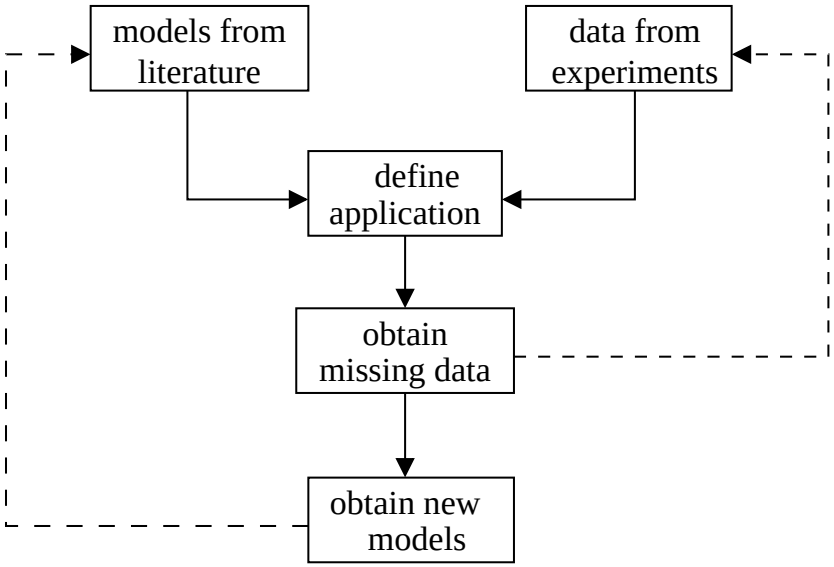
- numerical approximation of f using Taylor or Volterra series $\rightarrow$ local search
- scatter plots $\rightarrow$ machine learning (e.g., deep learning)
- efficient sampling of input space, e.g. Sobol's decomposition:

$$y = f(x_1, x_2, \ldots, x_n) = f_0 + \sum_i f_i(x_i) + \sum_{i<j} f_{i<j}(x_i, x_j) + \cdots$$

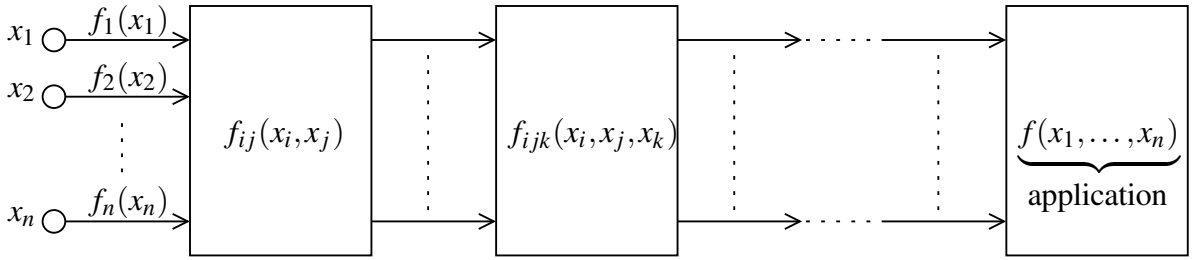
- output variance decomposition based on input-factors (e.g., ANOVA)
- meta-models
- optimization methods to find minimum of y (e.g. tabu search)

REVERSE-FORWARD MODELING

Typical scenario



Optimum decomposition of f

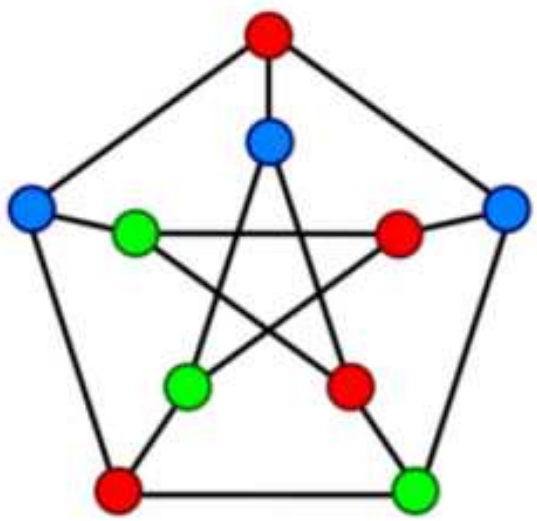


- find optimum path from leafs $x_1, x_2, \ldots, x_n$ to the root $f(x_1, \ldots, x_n)$ of this tree
- extension: multiple observations (a tree with multiple roots)

Problems over graphs

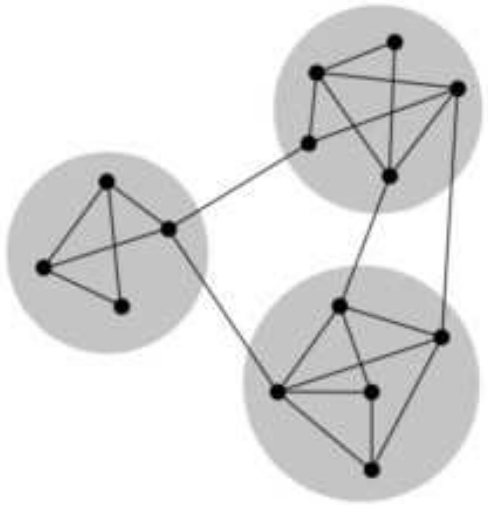
Subgraphs

- community detection
- maximum clique identification
- strongly connected components



Path

- minimum spanning tree
- maximum/perfect matching
- shortest path between a pair of nodes
- longest path in a graph (diameter)
- traveling salesman



Many other

- drawing and visualization
  → graph coloring
- graph search
- linear ordering of nodes (for acyclic graphs)
  → more generally, ranking and sorting of nodes
- maximum flow/minimum cut

Graph mining

Aim

- identify patterns and their frequency (i.e. support) within graph(s)

Patterns

- subgraphs: path, tree, cycle
- given a priori, or as a growing pattern during search
  → patterns with constraints (e.g. maximum degree, density, size, diameter)
- how to define small subset of representative patterns?

Search strategy

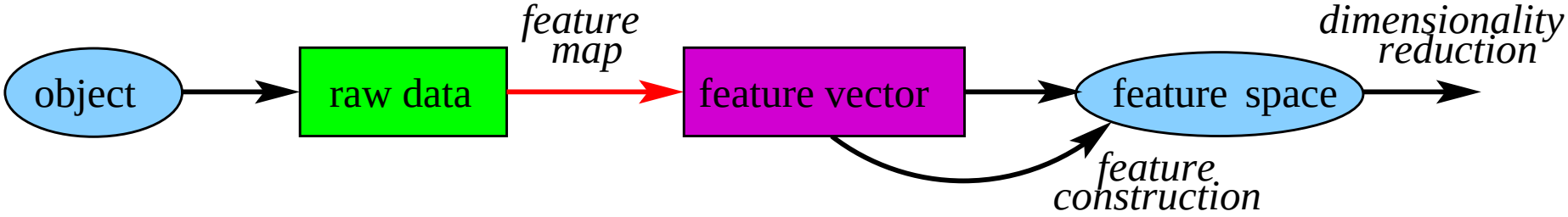
- depth first vs breadth first, complete vs. incomplete

Applications

- structure → function prediction (chemoinformatics, protenomics)
- graph clustering, classification, compression
- control flow testing (e.g. software)

Data features

Numerical representation of objects



$$\text{score}(\textit{object}) = \sum_i \text{weight}_i \cdot \text{feature}_i$$

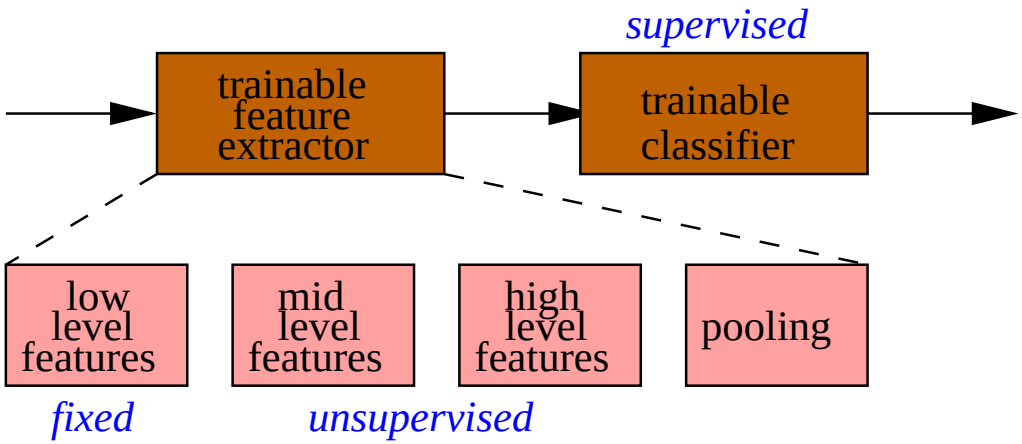
- feature extraction/selection hard-coded or they can be learned
- features facilitate Machine Learning
  → e.g. pattern recognition discovers relationships, i.e., patterns
- features reduce the amount of data and are sometime called predictors

Similarity function

- how similar are two (mathematical) objects?
- often related to distance metrics (e.g., l_p-norm) and inner-product

$$\text{similarity}(\text{feature}_1, \text{feature}_2) = \sum_i \text{feature}_{1i} \cdot \text{feature}_{2i}$$

Deep Learning



Hierarchical representations

- Image: pixel → edge → texton → motif → part → object
- Text: Character → word → word group → clause → sentence → story
- Speech: Sample → spectral band → sound → ... → phoneme → word

Linear classifier (most popular)

$$\text{Class} = \begin{cases} \text{Quantization}\left(\sum_i \text{Weight}_i \times \text{Feature}_i(\textit{Object})\right) \\ \text{Quantization}\left(\sum_i \text{Weight}_i \times \text{Kernel}(\text{Label}^{(i)}, \textit{Object})\right) \end{cases}$$

Why DL

- more efficient in representing complex functions than shallow architectures → better parametrization, less computations, less hardware

KERNEL

Replace

feature map: $\phi : \mathcal{X} \mapsto \mathcal{V}$

similarity function: $< \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') >_{\mathcal{V}} = \phi(\boldsymbol{x}) \cdot \phi(\boldsymbol{x}')$

with

$$\underline{K(\boldsymbol{x}, \boldsymbol{x}') = < \phi(\boldsymbol{x}), \phi(\boldsymbol{x}') >_{\mathcal{V}}}$$

- this is known as a *"kernel trick"*
- it avoids the need to learn (non-linear) feature map ϕ
  $\rightarrow$ inner-product in space $\mathcal{V}$ allows linear interpretation $\rightarrow$ linear algebra
- it may also avoid otherwise NP-difficult problem of similarity between complex (e.g. graph) data structures $\boldsymbol{x}$ and $\boldsymbol{x}'$

Graph kernels

- motivation: find similarity of (sub)graphs in polynomial time
  $\rightarrow$ subgraphs: walk, path, tree, cycle

Example: subtree kernel

- compare subtrees of given maximum height h from all pairs of nodes $v_1 \in G_1$ and $v_2 \in G_2$; then subtree kernel

$$K_{\text{tree},h}(G_1, G_2) = \sum_{\substack{v_1 \in G_1 \\ v_2 \in G_2}} K_h(v_1, v_2)$$

  suffers from tottering (visit the same nodes multiple times)

GRAPH TRAVERSABILITY

Objective

- in dense graphs, avoiding revisiting the same vertex is difficult

  → also to prevent traversing indefinitely

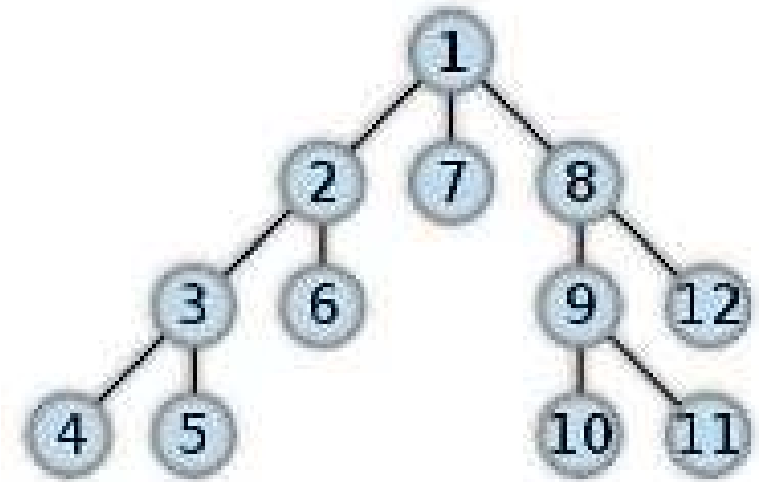
  → (adaptively) limit depth and breath considered

  → graph sampling (suffers from bias towards high degree nodes)

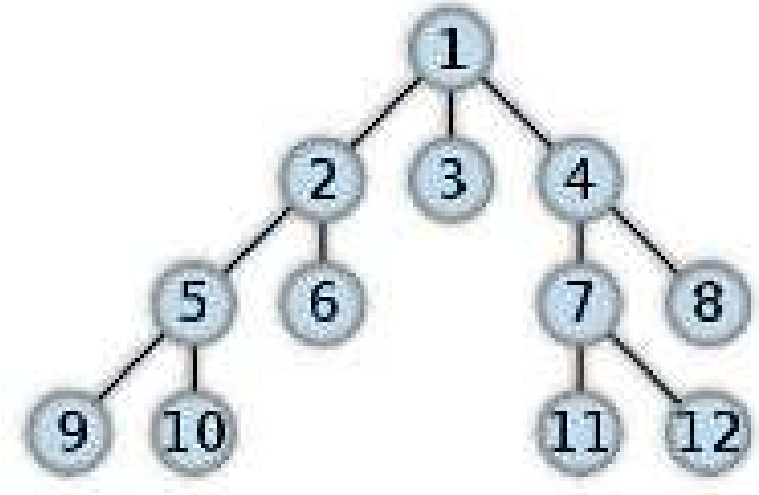
Applications

- find: connected components, spanning tree, shortest path
- discover graph structure on the fly
- compute maximum flow between two nodes
- serialization & deserialization of graphs

Depth-first search (DFS)



Breadth-first search (BFS)



Graphs to numbers

Variables

$$y = f(x)$$

y: dependent, target, explained, experimental, target, ...

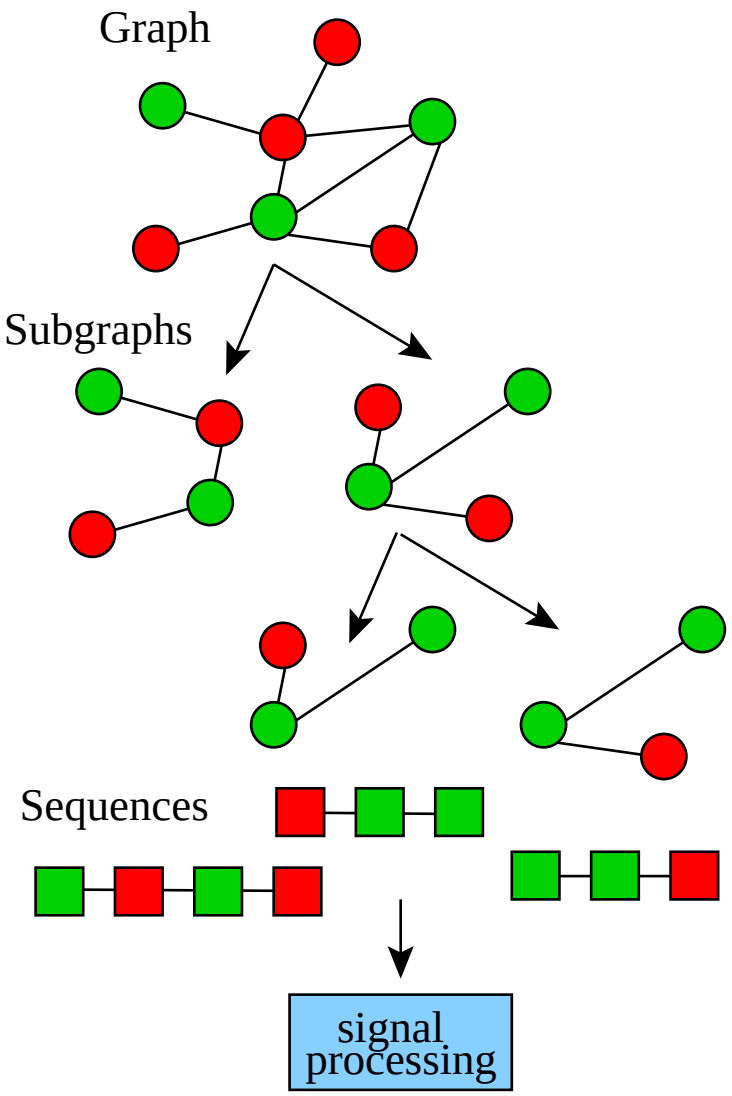
x: independent, explanatory, controlled, predictor, ...

Or

$x \in \mathcal{X}$: orderable feature (index, time, ...)
and note that given x, y is a number

Graph → sequence extraction

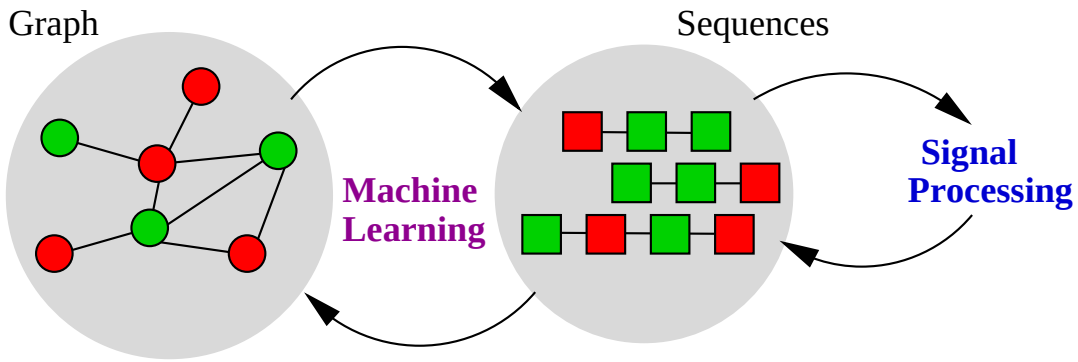
- middle step: subgraph
  → by constraints, or sampling
- output is sequence of nodes and/or edges
  → ordered data sequence
- a lot of signal processing tools available
  → filtering, estimation/detection, ...



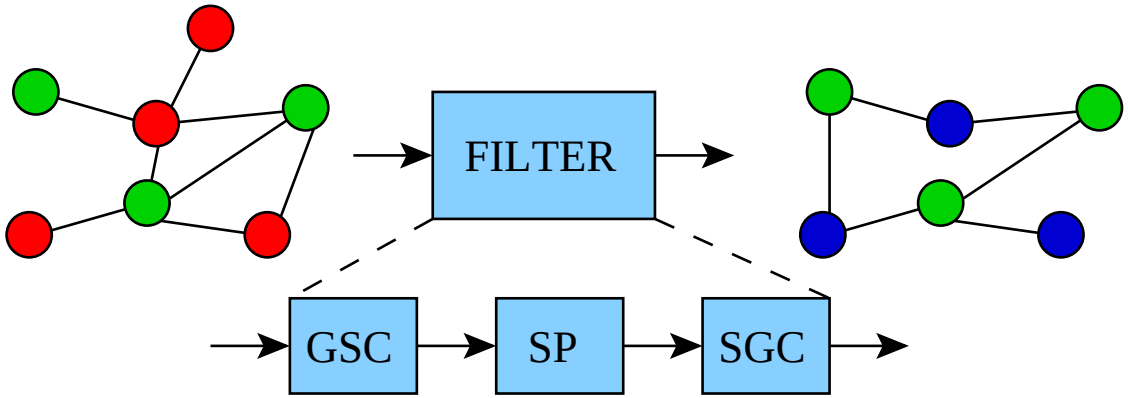
Numbers to graph

Inverse problem

- sequence $\xrightarrow{\text{Machine Learning}}$ graph



Graph filtering



- graph-to-sequences (GSC) and sequences-to-graph (SGC) converters
- algebra and arithmetic for mathematical complex structures
  $\rightarrow$ signal processing

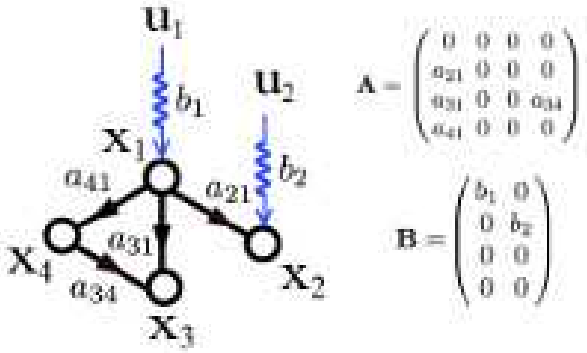
GRAPH OBSERVABILITY[1]

Linear graph

- $a_{ij}(t)$ is edge weight between nodes i and j
- x_i is weight of node i, so the internal state $\boldsymbol{x}(t) = [x_1(t), \ldots, x_n(t)]^T$
- if time-invariant, state-space description

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{A}\,\boldsymbol{x}(t) + \boldsymbol{B}\,\boldsymbol{u}(t) \\ \boldsymbol{y}(t) &= \boldsymbol{C}\,\boldsymbol{x}(t) \end{aligned}$$

... this is a linear MIMO system!

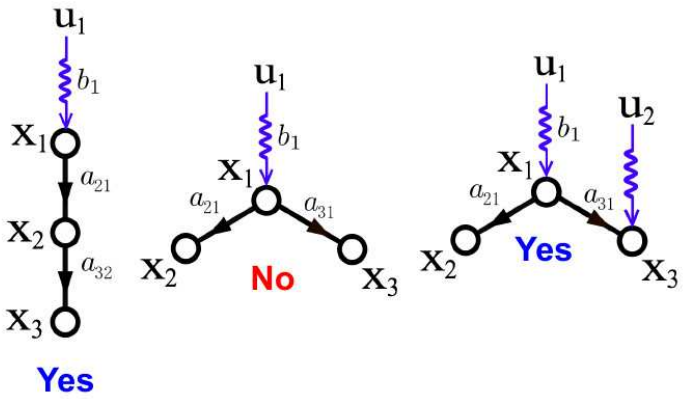


Observability

- a state (values of internal variables) can be obtained from finite observations
  $\rightarrow$ find state trajectory $\boldsymbol{x}(t)$ from any initial state $\boldsymbol{x}(0)$ to the current state
- strongly connected components (there is path between any two nodes)
  $\rightarrow$ have to observe at least one node from each SCC
- for linear model, can use maximum matching to find minimum # sensors
  $\rightarrow$ often much larger than # SCC (due to model symmetries)
- surprisingly, for non-linear model, sensors predicted by SCCs are necessary as well as sufficient (since model symmetries are rare)

[1] YY Liu, JJ Slotine, AL Barabási, Observability of complex systems, PNAS 2013.

GRAPH CONTROLABILITY[1]



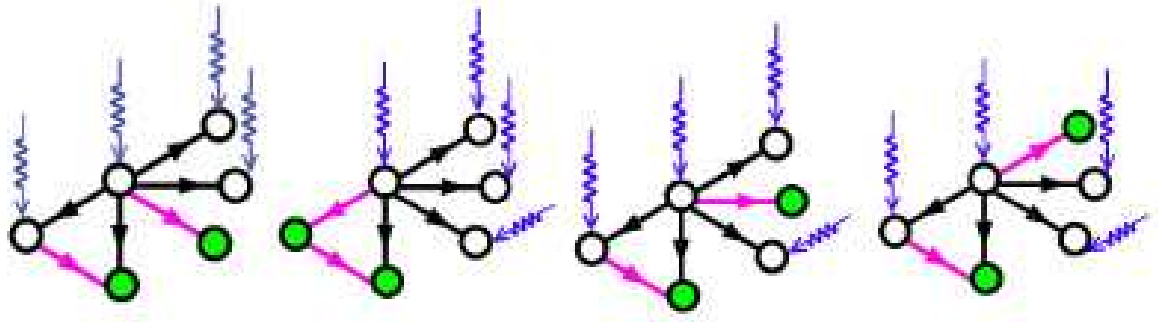
Kalman's condition

- controlability matrix $\boldsymbol{D}$ must be full rank i.e. $\text{rank}\boldsymbol{D} = n$ where $n = |\boldsymbol{x}|$

$$\boldsymbol{D} = [\boldsymbol{B}, \boldsymbol{AB}, \boldsymbol{A}^2\boldsymbol{B}, \ldots, \boldsymbol{A}^{n-1}\boldsymbol{B}]$$

Driver nodes

- for linear networks, maximum matching again helps



- surprisingly, driver nodes tend to avoid hubs
  → average degree of driver nodes is smaller than average degree of graph
  → # driver nodes mainly determined by degree distribution

"Sparse and heterogeneous networks are harder to control than dense and homogeneous networks."

[1] YY Liu, JJ Slotine, AL Barabási, Controllability of complex networks, Nature 2011.

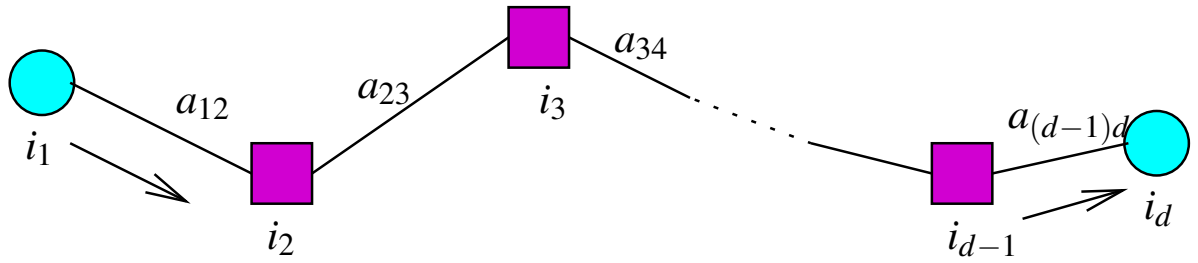
GRAPH TOMOGRAPHY

Linear graph

- assume a graph with the weight $a_{ij} \in \mathcal{R}$ between nodes i and j
- let path $\mathcal{P} = \{(i_1 i_2), (i_2 i_3), \ldots, (i_{d-1} i_d)\}$ from node i_1 to node i_d
  $\rightarrow$ accumulated weight along the path $\mathcal{P}$ is $a_{i_1 i_d} = \sum_{(ij)\in\mathcal{P}} a_{ij}$
  $\rightarrow$ in matrix notation

$$\boldsymbol{y} = \boldsymbol{P} \cdot \boldsymbol{a} + \boldsymbol{w}$$

where rows of binary matrix $\boldsymbol{P} \in \{0,1\}^{m \times n}$ correspond to each measured path, $\boldsymbol{a}$ is graph adjacency vector, and m denotes the number of probes



Multiplicative weights

- if $a_{i_1 i_d} = \prod_{(ij)\in\mathcal{P}} a_{ij}$, we can assume model

$$\log y_{i_1 i_d} = \sum_{(ij)\in\mathcal{P}} \log a_{ij} + w$$

GRAPH TOMOGRAPHY (2)

Task 1

- given measurements $\boldsymbol{y}$, find minimal path matrix $\boldsymbol{P}$ to recover k weights in $\boldsymbol{a}$
  $\rightarrow$ this assumes that the graph structure is known
- the path matrix $\boldsymbol{P}$ is minimal if either the sum-length of all probing paths considered is minimum, or if the longest among these paths is minimized
- trivial (non-minimal) design: measure all weights a_{ij} one by one
  $\rightarrow$ determining all paths of a general graph is NP-hard
  $\rightarrow$ in practice, only a subset of nodes may be used for I/O or as gateways

Task 2

- knowing the graph structure, and given subset of paths $\boldsymbol{P}$ and the corresponding measurements $\boldsymbol{y}$, recover k graph weights in $\boldsymbol{a}$
  $\rightarrow$ $\boldsymbol{P}$ may not be large enough to enable compressive sensing of $\boldsymbol{a}$
  $\rightarrow$ some weights in $\boldsymbol{a}$ may be calculated if overdetermined system $(m \geq k)$
- different strategy:
  $\rightarrow$ determine weights change $\Delta\boldsymbol{a}$ if nominal weights $\boldsymbol{a}$ are known

$$\boldsymbol{y} = \boldsymbol{P} \cdot (\boldsymbol{a} + \Delta\boldsymbol{a}) + \boldsymbol{w} = \underbrace{\boldsymbol{P} \cdot \boldsymbol{a}}_{\text{known}} + \boldsymbol{P} \cdot \Delta\boldsymbol{a} + \boldsymbol{w}$$

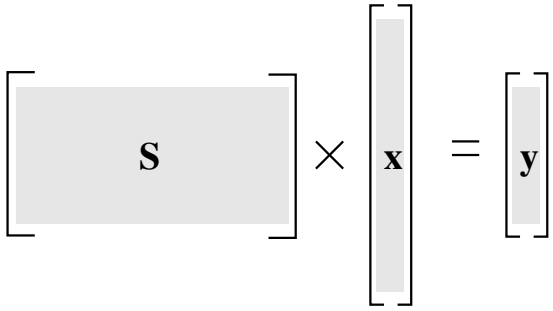
assuming k'-sparse change vector $\Delta\boldsymbol{a}$ with $k' \ll k$

Compressive sensing

Linear sensing (in discrete time)

$$\boldsymbol{y} = \boldsymbol{S} \cdot \boldsymbol{x} + \boldsymbol{w}$$

$\boldsymbol{y} \in \mathcal{R}^m$	:	compressed/observed signal
$\boldsymbol{S} \in \mathcal{R}^{m \times n}$	:	compressive/sensing matrix
$\boldsymbol{x} \in \mathcal{R}^n$	:	desired signal
$\boldsymbol{w} \in \mathcal{R}^m$	:	observation noise



Task

- given $\boldsymbol{y}$ and $\boldsymbol{S}$, and norms l_p and l_q, obtain approximation $\boldsymbol{x}^*$ of $\boldsymbol{x}$ such that

$$\|\boldsymbol{x}^* - \boldsymbol{x}\|_p \leq C(k) \min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_q$$

  over all k-sparse signals $\tilde{\boldsymbol{x}}$ and for $m \ll n$
- the best $\boldsymbol{x}^*$ contains k largest abs values of $\boldsymbol{x}$, so if $\boldsymbol{x}$ is also k-sparse, then $\boldsymbol{x}^* = \boldsymbol{x}$ and the recovery is exact
- typically assumed norms

$$\begin{aligned} \|\boldsymbol{x}^* - \boldsymbol{x}\|_2 &\leq C \min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_2 \\ \|\boldsymbol{x}^* - \boldsymbol{x}\|_1 &\leq C \min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_1 \\ \|\boldsymbol{x}^* - \boldsymbol{x}\|_2 &\leq (C/\sqrt{k}) \min_{\tilde{\boldsymbol{x}}} \|\tilde{\boldsymbol{x}} - \boldsymbol{x}\|_1 \end{aligned}$$

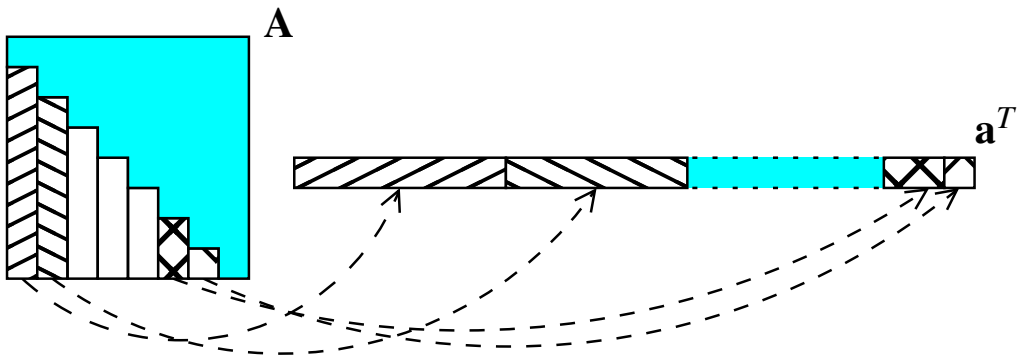
Questions

- what is minimum m for given n and k? Typically, $m \geq C \cdot k \cdot \log n$
- how to find $\boldsymbol{S}$? RIP: $\|\boldsymbol{x}\|_2 \approx \|\boldsymbol{A}\boldsymbol{x}\|_2$

COMPRESSIVE SENSING FOR GRAPHS

Adjacency vector

- undirected simple graph $G(V,E)$ (no loops) has at most $n = \frac{1}{2}(|V|^2 - |V|)$ edges
  $\rightarrow$ define adjacency vector $\boldsymbol{a} \in \mathcal{R}_+^n$ from adjacency matrix $\boldsymbol{A} \in \mathcal{R}_+^{|V|\times|V|}$



Graph compression

- if graph $G(V,E)$ is sparse, adjacency vector $\boldsymbol{a}$ is k-sparse, and $k = |E| \ll n$
- define compression matrix $\boldsymbol{S}$ for known k-sparse vector $\boldsymbol{a}$ in $\{0,1\}^{n\times n}$ or $\mathcal{R}_+^{n\times n}$
  $\rightarrow$ dense random $\boldsymbol{S}$ may be a good choice

Graph decompression

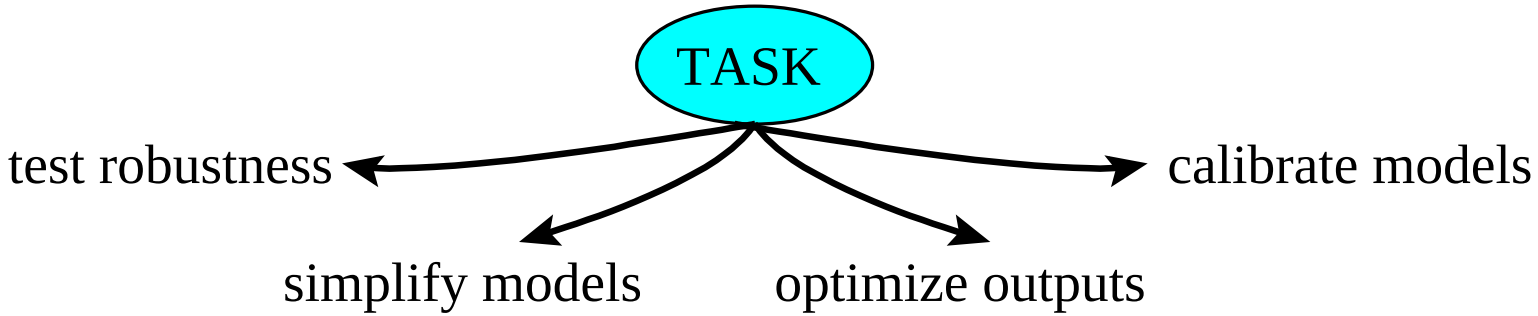
$$\text{no noise:} \quad \min \|\boldsymbol{a}^*\|_1 \quad \text{s.t.} \quad \boldsymbol{S}\,\boldsymbol{a}^* = \boldsymbol{S}\,\boldsymbol{a}$$

$$\text{with noise:} \quad \min \|\boldsymbol{a}^*\|_1 \quad \text{s.t.} \quad \|\boldsymbol{S}\,\boldsymbol{a}^* - \boldsymbol{y}\|_2 \leq \epsilon$$

Network sensitivity analysis

Objectives

- quantify effect of uncertainties in model parameters (not in model structure)
- OR, identify the most influential inputs and parameters (factors) of a model



General challenges

- local vs. global, high-dimensional space, computational demand

Network specifics

- optimizing edge/vertex weights
  → (un)constrained by zeros in adjacent matrix $\boldsymbol{A}$ (i.e. by network structure)
- difficult to find the feasibility region (especially for changes in structure)
- in some applications, number of network nodes may be optimized

Network sensitivity analysis[1] (2)

System: $Y = f(\boldsymbol{X})$

OTA: $\left|\frac{\partial Y}{\partial X_i}\right|_{\boldsymbol{X}_0}$ or $\frac{X_{i0}}{Y_0}\left|\frac{\partial Y}{\partial X_i}\right|_{\boldsymbol{X}_0}$

effect index: $S_i = \frac{V_i}{\text{var}[Y]}$

$$V_i = \text{var}_{X_i}\Big[\text{E}_{\sim X_i}[Y|X_i]\Big]$$

ANOVA decomposition:

$$\text{var}[Y] = \sum_{i=1}^{d} V_i + \sum_{i<j}^{d} V_{ij} + \cdots + V_{12\ldots d}$$

$$V_{ij} = \text{var}_{X_{ij}}\Big[\text{E}_{X_{\sim ij}}\Big[Y|X_i, X_j\Big]\Big] - V_i - V_J$$

Network specifics

$$Y = f(\boldsymbol{X}, \boldsymbol{A}) \quad \longrightarrow \quad \left|\frac{\partial Y}{\partial \boldsymbol{A}_{ij}}\right|_{\boldsymbol{X}_0}$$

where $\boldsymbol{X}$ are network inputs and $\boldsymbol{A}$ are network weights to be optimized

Dynamic systems

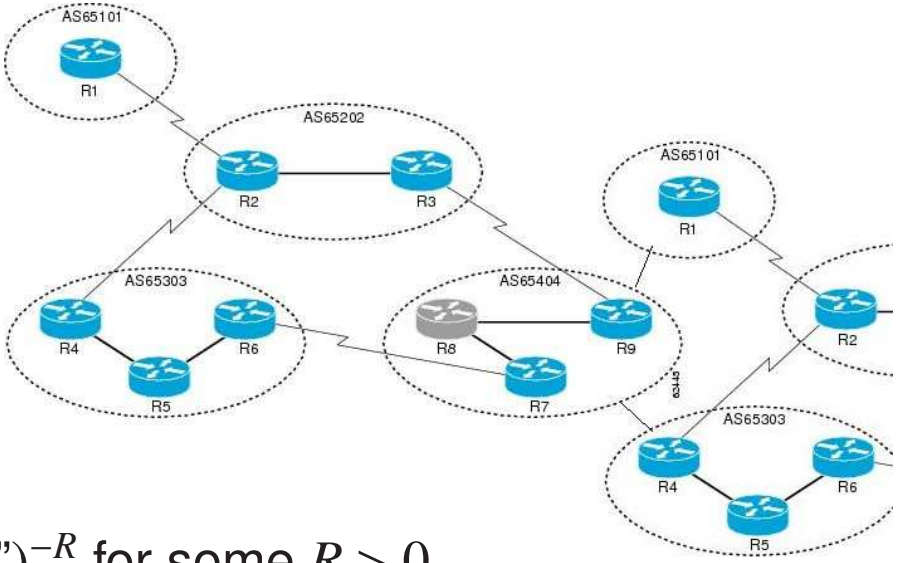
$$Y(t) = f(\boldsymbol{X}(t), \boldsymbol{A}(t), t)$$

e.g. maximize average output $\boldsymbol{A}^* = \text{argmax}_{\boldsymbol{A}}\,\text{E}[\,|Y(t)|\,]$

[1] A Saltelli et al., Global Sensitivity Analysis, Wiley, 2008.

Modeling the Internet[1]

Autonomous systems (AS)



Power-laws in the Internet

1. "node degree" $\sim$ ("node index")$^{-R}$ for some $R > 0$
2. "% of nodes of degree at least d'' $\sim d^{-1/R}$
3. "i – th eigenvalue of connectivity matrix" $\sim i^{-1/(2R)}$
4. "number of node pairs within at most h hops" $\sim (|V|+2|E|)\,h^H$ for some $H > 0$ and if h is less than the diameter of network $N(V,E)$

Topology properties

1. rich-club structure with connectivity index for r largest nodes $\sim r^{-\gamma}$ where $\gamma = 1.1$ for AS level and $\gamma = 1.8$ for router level Internet
2. the neighbors of hubs have few connections (80% less or equal to 3)

[1] G Chen, X Wang, X Li, Fundamentals of Complex Networks, Wiley, 2015.

Modeling the Internet (2)

Growth of the Internet

- characteristics almost unchanged:
  - small average path length
  - large clustering coefficient
  - power-law distributions of node degrees
- other typical behavior:
  - small degree nodes much higher probability of change (add/delete/rewire)
  - birth rate of nodes and edges usually larger than their removal rate
  - capacity driven both by addition of links and their upgrades
  - router level topology kept confidential by ISPs
  - Internet density R strongly correlated with population density P as $R \sim P^{\alpha}$

Internet topology generators

- should account for all these properties i.e. fit data and measurements
- over time, the Internet is getting more complex (and does not behave as a random network) while our understanding also improves
- most generators iterative $\rightarrow$ nodes/edges added sequentially
- regional networks having high clustering are interconnected sparsely

Modeling the Internet (3)

Random graph network topology generator

- start with N nodes on finite lattice
- add edges between randomly selected nodes with certain probability
  $\rightarrow$ this generates typical random graph with small connected component

Tiers topology generator

- focus on hierarchical structure with WAN-MAN-LAN sub-networks
- generation from top level down
- pre-defined redundancy of edges at all hierarchical levels

Transit-stub topology generator

- again 3-layer structure generated top-down
- group of parameters for number of nodes and another for number of edges

Internet connectivity generator

- focus on power-law out-degree node distribution while fitting real data
- use preferential attachment with empirically calculated weightings
- exploit extrapolation of nodes at AS level

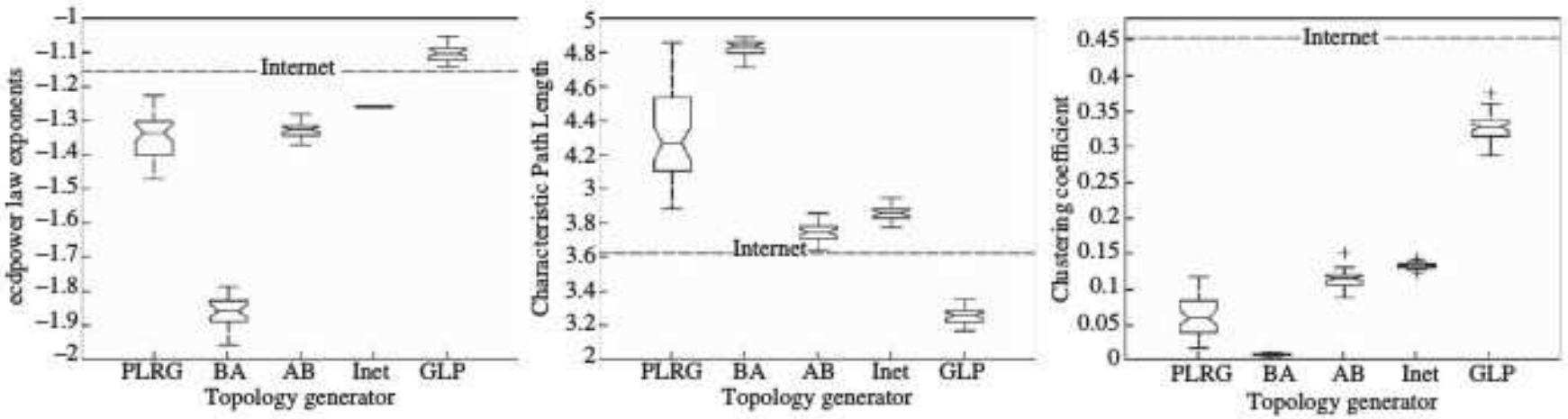
Modeling the Internet (4)

BRITE model

- focus on 3 design principles:
  - representative: match the topology and node degrees
  - inclusive: incorporate/reuse parts from other generators
  - interoperability: models for import by software packages (e.g. NS2)
- flexibility in specifying # nodes added at once, and how to connect them

GLP model

- further improves BA model
- again uses a probability p to add new nodes and decide connections
  $\rightarrow$ addition of new nodes and edges is independent



Modeling the Internet (5)

PFP model

- focus on rich-club phenomenon and uses non-linear preferential attachment

T_{ANG} model

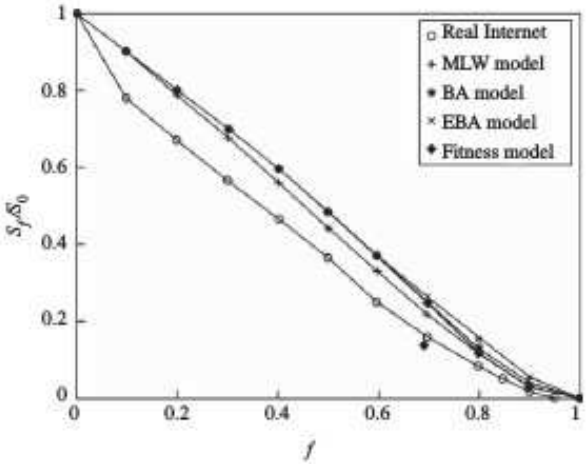
- modifies preferential attachment (degrees $k_i \mapsto k_i^{1+\epsilon}$) and accounts for non-equivalent nodes (e.g. ISP vs. end-users)

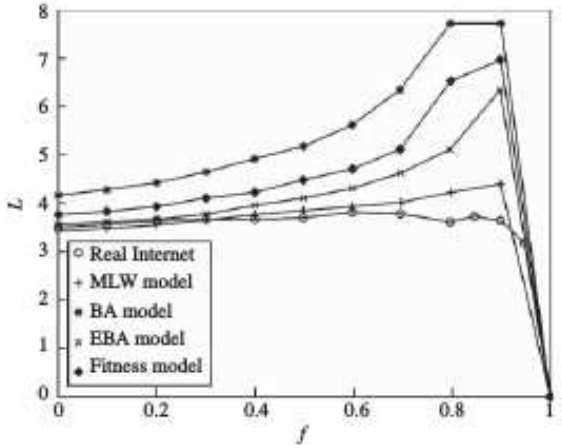
Multi-local world model

- addition of an AS also affects connections among existing nodes
  $\rightarrow$ rewiring/adding/deleting
- at AS level, rates of edges creation and deletion are comparable

Comparisons

	BA model	EBA model	Fitness model	MLW model	Real Internet (15 May 2005)
N	21 999	21 999	21 999	21 999	21 999
C	0.003	0.009	0.012	0.238	0.457
L	4.154	3.533	3.731	3.410	3.493
γ	3	2.2	2.26	2.2	2.2





Modeling the Internet (6)

HOT model

- cooperative nodes → preferential attachment → power-law degree distribution → scale-free networks
- completing nodes → selfishness and optimize their attachments
  → trade-offs among competing objectives
  → a new node added as to minimize a cost function

Deficiencies of these models

- they generate static topology while the Internet topology is highly dynamic
- all links treated as being equal (in practice, huge differences in capacities)
- partly modeled or completely ignored:
  - traffic dynamics → $\sigma_{\text{traffic}} \sim < \text{traffic} >^{\alpha}$
  - edge weights (e.g. $k_i^{\theta} k_j^{\theta}$) and nodes strength (e.g. $\sum_{ij} k_i^{\theta} k_j^{\theta}$)
  - routing costs (e.g. delays)

GRD model

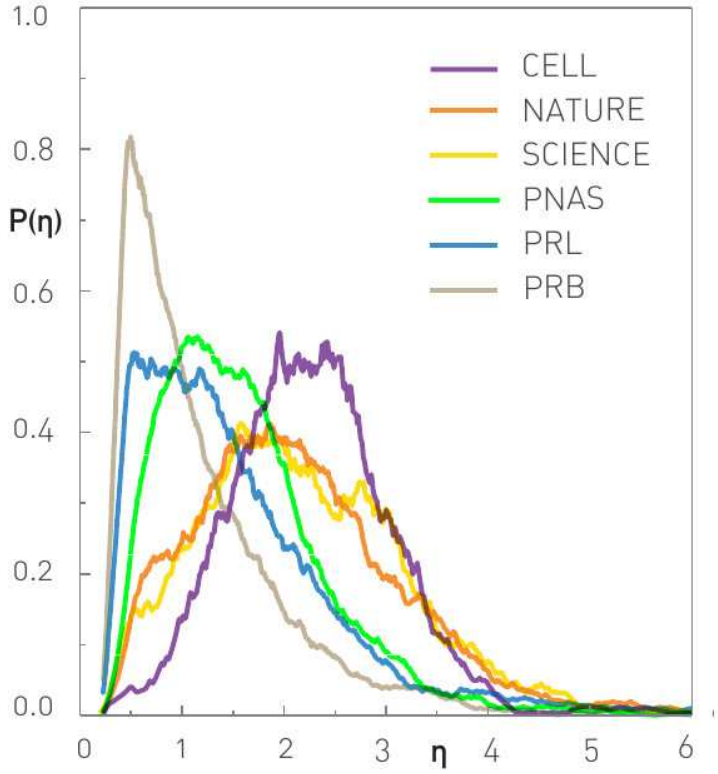
- random walk modeling of traffic fluctuations on weighted networks
  → random walk follows preferential attachment probabilities
  → traffic fluctuations on edges as well as nodes

Modeling the Internet (7)[1]

Bianconi-Barabási (BB) model

- focus on fitness, e.g. ability of a node to make connections long-lasting
- fitness η of a node is assigned randomly from a fitness distribution
  $\rightarrow$ node fitness does not change over time
- preferential attachment probabilities $\Pi_i = \frac{\eta_i k_i}{\sum_i \eta_i k_i}$
  $\rightarrow$ e.g., a latecomer with a great product acquires customers faster than the competition (often seen on Facebook)

- in BA model, the earlier the node joins, the more connections it will acquire
- if $\eta_i = \eta \ \forall i$, BB model becomes BA model
- if fitness of a node can be measured, we could predict which website, research paper etc. will quickly gain visibility
  $\rightarrow$ fitness is a network collective perception (not opinion of an individual)
  $\rightarrow$ fitness distribution can be determined from evolution of node degrees



[1] AL Barabási, Network Science, Evolving Networks, 2014.

Network Game Theory[1]

Prisoner's dilemma (PD)

- 2-player, 2-strategy game

$$\text{player 1} \begin{cases} \text{C} \\ \text{D} \end{cases} \overbrace{\begin{bmatrix} R & S \\ T & P \end{bmatrix}}^{\text{player 2}} \quad \begin{matrix} \text{C} & \text{D} \end{matrix}$$

C:	cooperator	R:	reward
D:	defector	S:	sucker's payoff
		T:	temptation to defect
		P:	punishment

Optimum strategy

- if $T > R > P > S$, then be selfish (Nash equilibrium)

PD on networks

- neighboring players are comparing and learning their successful strategies
  $\rightarrow$ replicator dynamics

$$\Pr\left(S_i \leftarrow S_j\right) = \underbrace{f(P_j - P_i)}_{\text{payoffs difference}} \overset{\text{Fermi}}{=} \left(1 + \exp[-(P_j - P_i)/\kappa]\right)^{-1}$$

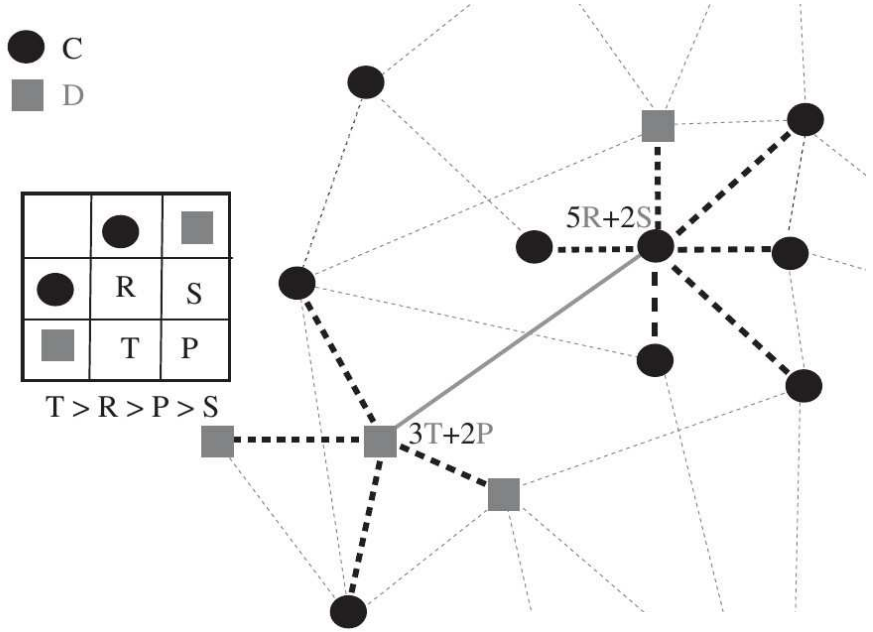
$$\kappa \begin{cases} = 0 & \text{rational decisions, perfect learning of neighbor's strategy} \\ > 0 & \text{irrational, may learn worse strategy from neighbor} \end{cases}$$

[1] G Chen, X Wang, X Li, Fundamentals of Complex Networks, Wiley, 2015.

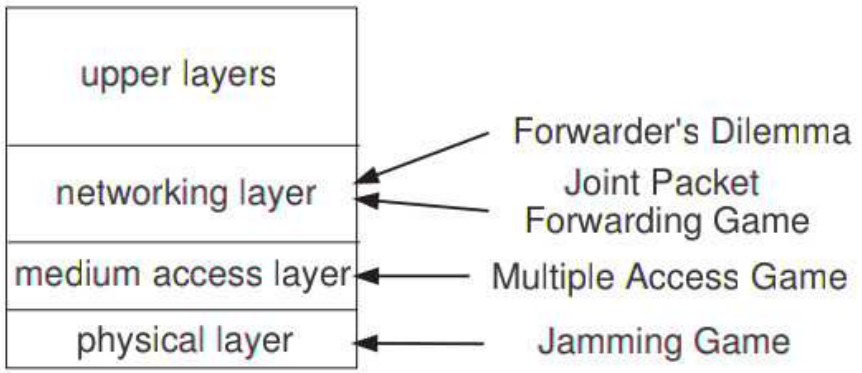
Network Game Theory (2)

Phenomena

- distinctive behaviors on different networks and various $\Pr\left(S_i \leftarrow S_j\right)$
  $\rightarrow$ cooperation may emerge or cease $\rightarrow$ diffusion of behaviors



Games in wireless networks[1]

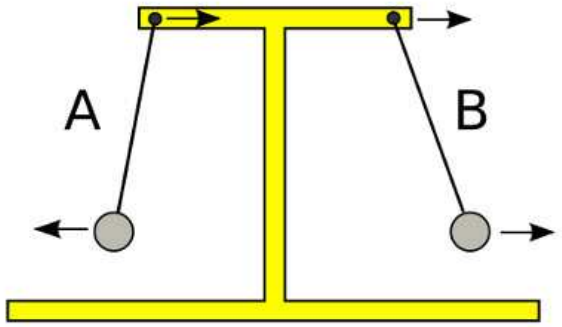


[1] M Felegyhazi, JP Hubaux, Game Theory in Wireless Networks: A Tutorial, 2007.

Network synchronization

Synchronization phenomenon

- spontaneously appearing among (even weakly) coupled components
- e.g. the Internet routers tend to get synchronized[1] → abrupt phase transition



Mathematical model

$$\dot{\boldsymbol{x}}_i = f(\boldsymbol{x}_i) + C \sum_{\substack{j=1 \\ i \neq j}}^{N} a_{ij} \boldsymbol{H}(\boldsymbol{x}_j) \qquad (1)$$

$\boldsymbol{x}$: inner state vector
$f(\cdot)$: non-linear dynamics
C: coupling strength
a_{ij}: element of adjacency matrix $\boldsymbol{A}$ (outer coupling)
$\boldsymbol{H}(\boldsymbol{x}_j)$: inner coupling, if linear, $\boldsymbol{H}(\boldsymbol{x}_j) = \boldsymbol{H}\boldsymbol{x}_j$

Definition

- network becomes fully synchronized when $\lim_{t\to\infty} \left\| \boldsymbol{x}_i(t) - \boldsymbol{x}_j(t) \right\| = 0$ for $\forall i, j$
- if and how that happens depend on f, C, $\boldsymbol{H}$ and $\boldsymbol{A}$ when solving (1) above; e.g. for highly dynamic f, the network is unstable and hardly synchronizes

[1] S Floyd, V Jacobson, The synchronization of periodic routing messages. Computer Comm. Review, 1993.

Network synchronization (2)

Small-world and scale-free networks

- SW construction: edges (couplings) are changed with probability p
- BA construction: edges (couplings) added by preferential attachment rule
- periodically calculate the smallest non-zero eigenvalue of Laplace matrix $\boldsymbol{L}$
  $\rightarrow$ indicates synchronizability
- generally, most large scale networks are synchronizable

Optimum network

- modify WS and BA network constructions, so that at each step, the network synchronizability is maximized $\rightarrow$ yields significant improvement
- such modified construction creates a multi-center network: a few nodes connected to large hubs while majority of nodes have little connectivity
  $\rightarrow$ these networks are very vulnerable to attacks (destroying nodes or edges)
- recent observation: removing some edges may increase synchronizability as well as robustness

Metrics for the Digital Economy

New developments

Application programming interfaces (APIs)

- standardized, controlled, trusted, secure and wholesale access to services

Digital platforms

- universal marketplaces connecting digital producers with consumers

Digital business models

- often cloudified and formed as XaaS (anything as a service)

Digital ecosystems

- complete solutions and strategies for delivering digital services
- orchestrate architectures, infrastructures, interfaces, policies, and service definitions
- should be trusted, highly autonomous and self-configuring

Operations and business support systems (OSS and BSS)

- various integrated policy-driven autonomous systems supporting key processes and business models in enterprises

New developments (2)

5G networks

- different application areas (verticals) with vastly different requirements
- enhanced mobile broadband (eMBB)
- ultra-high reliability and low latency (uRLLC) networks
- massive machine-type communications (mMTC)

Digital roadmap

- strategy towards digital monitoring and control of processes and operations
- communication service providers (CSPs)

Internet of Everything (IoE)

- interconnect digital platforms, IoT and everything else in different verticals
- key driver is customer experience and efficient utilization of resources

Edge (fog) computing

- distributing computing and digital content centers at network edges

Cyber-physical systems

- exploit connectivity and digitalization to significantly enhance efficiency of physical systems and improve business and customer experience

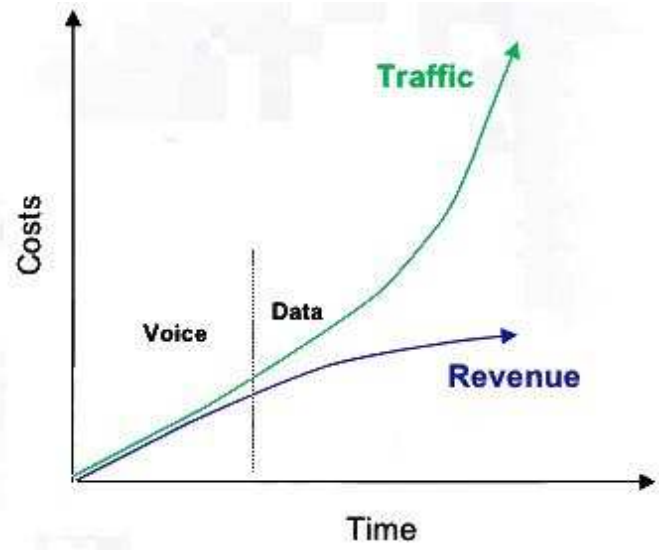
New developments (3)

New apps and markets

- over-the-top (OTT) applications
- new businesses and consumer markets (B2C and B2B)

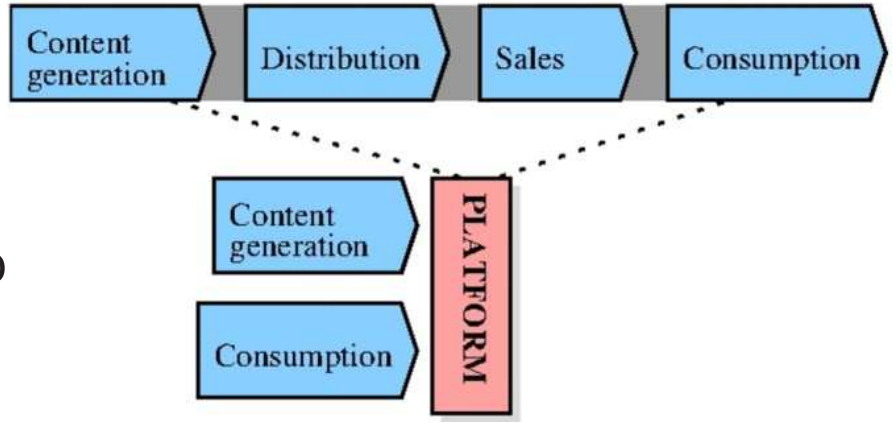
Average revenue per user (ARPU)

- constantly decreasing, competition with digital providers



Digital transformation

- changes in business rather than technology
- omni-channel customer interactions
- agility, cognition, innovations, rapid prototyping
- softwarization, virtualization, modularizatio
- micro-services, micro-data, APIs & clouds
  $\longrightarrow$ reduce time to market, OTT services



Metrics and Measurements

Key characteristics

1. Accuracy: measurement errors and biases to be within acceptable limits
2. Validity: measurements and their evaluations to be checked for correctness
3. Feasibility: measurements to be collected as often as desired
4. Robustness: measurements quality not be affected by changing conditions
5. Efficiency: measurements not to consume too much of system resources
6. Desirability: measurements collected are required for design and operation
7. Viability: measurements can clearly provide measurable benefits

Measuring digital maturity

Key performance considerations

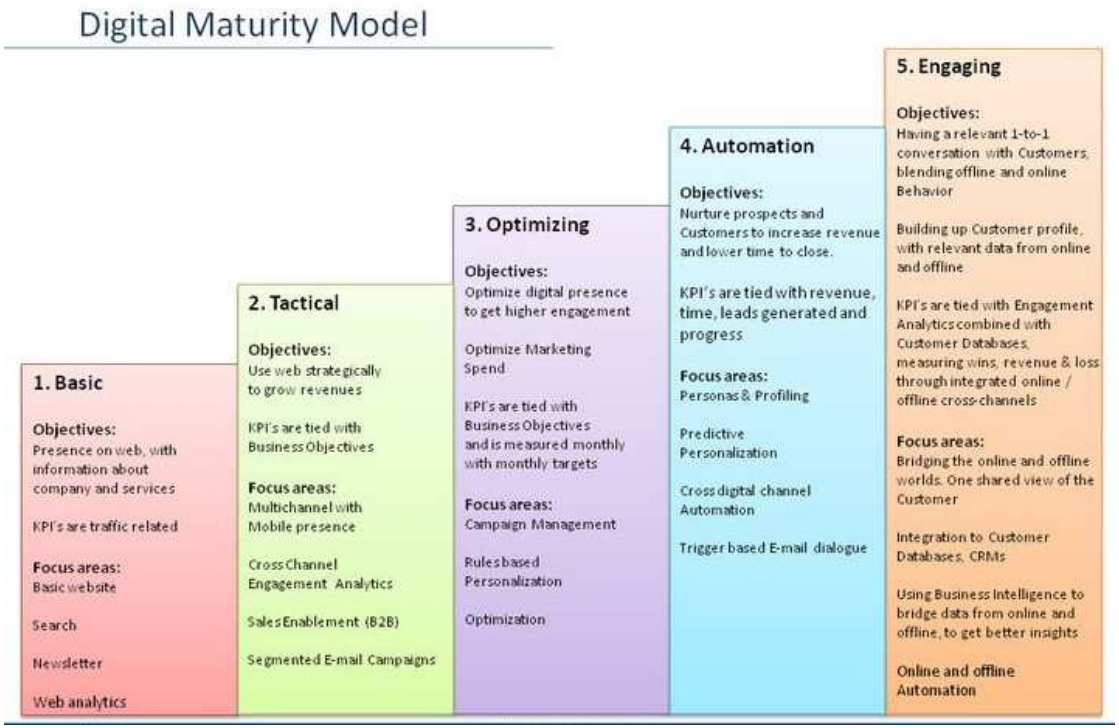
- revenues and profits, customer experience, and operational efficiency

Digital maturity models

- assesses readiness and effectiveness of the enterprise on its digital journey
- digital strategy, understanding of customers, human resources and other assets, processes, operations, and required technology

TM Forum model:

- 5 dimensions: customer, strategy, technology, operations, and organization
- 28 sub-dimensions and 175 criteria
- balancing costs and benefits is important



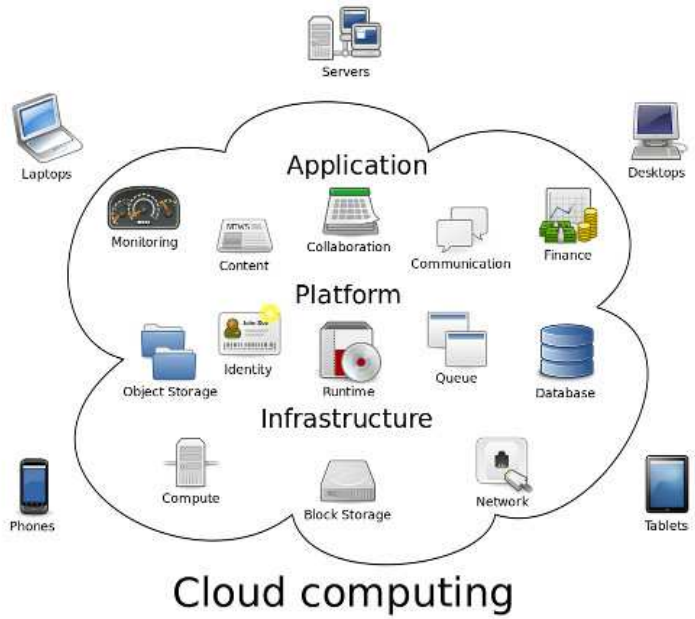
Metrics for computing systems

Mostly clouds

- security and demand policies, admission rules
- shareholders: infrastructure and application providers, 3rd party application and content providers, and service consumers
- technological and business performance

Cloud models

1. Infrastructure as a Service (IaaS)
2. Architecture as a Service (AaaS)
3. Platform as a Service (PaaS)
4. Anything as a Service (XaaS)



Cloud implementations

- Amazon Web Services (AWS)
- Google Cloud Platform (GCP)
- Microsoft Azure

Pricing models

- pay-as-you go
- flat-rate

Metrics for clouds

Service and system availability

- percentage of time the system is operational
- equivalently, average downtime over given time period, or average time between failures, or average time to recovery

Response reliability

- fraction of satisfactorily handled requests or service outcomes

Response time

- average time until the response is received after a request
- greatly affected by demand, so indicates scalability

Number of security threats and incidents in given time period

- service attractiveness for unauthorized use and level of security detection

Throughput or bandwidth

- the number of transactions or requests handled per unit of time
- important for real-time or large scale systems

Metrics for clouds (2)

Capacity or maximum utilization

- ability to concurrently handle most of workload requests without delay
- the maximum available computing power or storage space for a single user
- or, the total number of processors, total memory available etc.
- or, as scalability, the maximum number of requests served at any given time

Cost per request/workload unit/user

- includes all supporting, operational and business processes, and other recurring costs to provide agreed level of service
- for well designed system, decreasing over time to drive revenues (profits)

Return on invested capital (ROIC)

- number of years until total profit covers accumulated costs (CapEx & OpEx)

Market share

- should be growing to ensure the business viability

Metrics for websites/Web analytics

Website traffic

- longer or shorter-term trend of online visits
- visitors: new, repeated, returning, unique

Traffic sources

- visit origin: via search engine, another referral, website, or directly
- used for search engine optimization (SEO)

Bounce rate

- percentage of visitors only seeing single page, not exploring other pages
- on social media, the number of shares is often considered

Conversion rate

- ratio of unique visitors to the number of conversions
- conversions: visit recommended site, subscribe to service, purchase product
- derived metrics: value per visit and cost per conversion

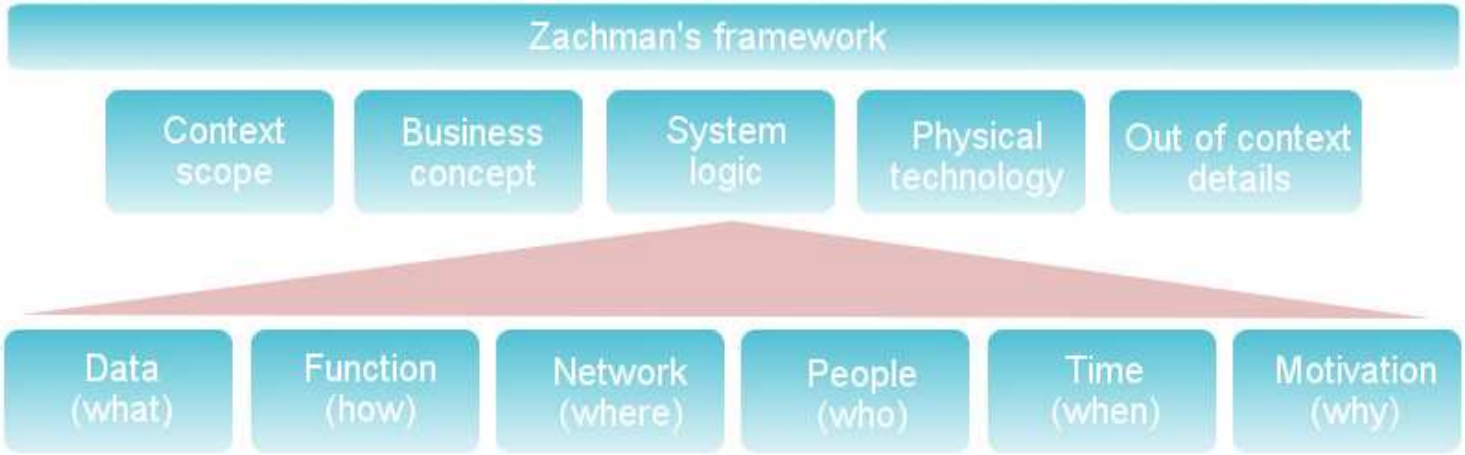
Other metrics

- visit duration, click through, exit page rate

Metrics for digital platforms

Zachman framework

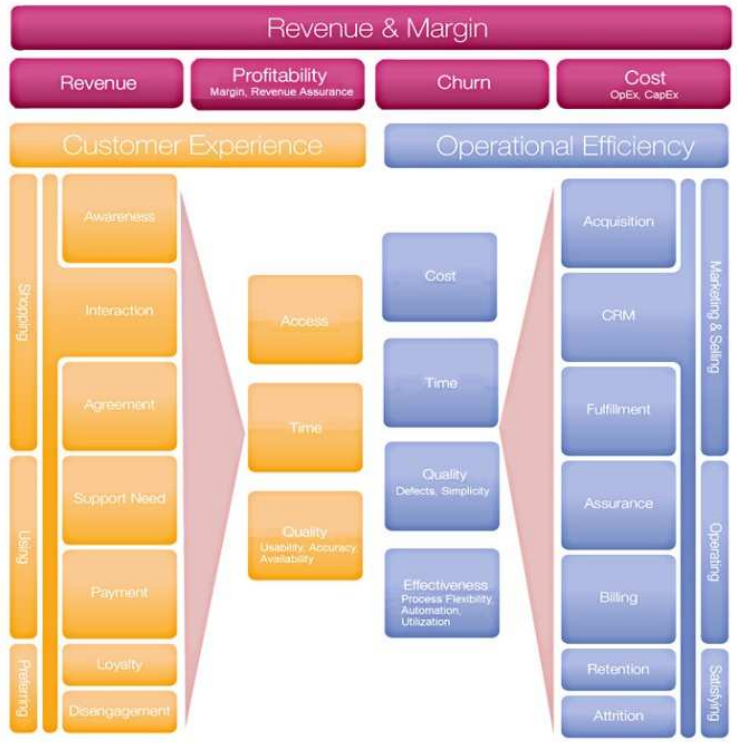
- popular enterprise modeling methodology proposed in 80's
- systematic and consistent approach to capture stakeholders' interests
- 2D matrix: 5 principal viewpoints, 6 model attributes
- however, insufficient for digital enterprises (way too complex)



Metrics for digital platforms (2)

TM Forum framework

- need to systematically record, categorize, update, search, maintain about 3,000 metrics
- standards and best practices to assess and optimize digitalized businesses
- service oriented, support automation of business processes
- metrics categories: business, customer experience management, cyber operations, fraud management, cable operations



Aims of TM Forum framework

- drive innovation, reduce risks and costs, shorten time to markets

Metrics have many attributes

- ID, business driver, capability, reporting, accuracy, responsible entity, capture period, name, value type, value range, metric type
- objective, purpose, level, state, topic, mathematical formula, symbols used, primary usage, domain, related metrics

Metrics for broadband networks

Broadband networks

Broadband service

- OECD definition: speed at least 256 kbit/s (good for WWW and VoIP)
- many countries adopt mandatory minimum speed (>256 kbit/s)
- positive correlation with economic and social development

OECD speed tiers

- less than 1.5/2 Mbit/s, 1.5/2-10 Mbit/s, 10-25/30 Mbit/s, 25/30-100 Mbit/s, 0.1-1 Gbit/s, and above 1 Gbit/s

OECD metrics categories

- broadband availability metrics and mappings
- broadband infrastructure investment metrics
- broadband performance metrics
- broadband competition metrics

Connection speed

- most important key performance indicator (KPI)
- wireline (fixed) vs. wireless (mobile), voice vs. data services, monthly data allowances, households/individual vs. businesses

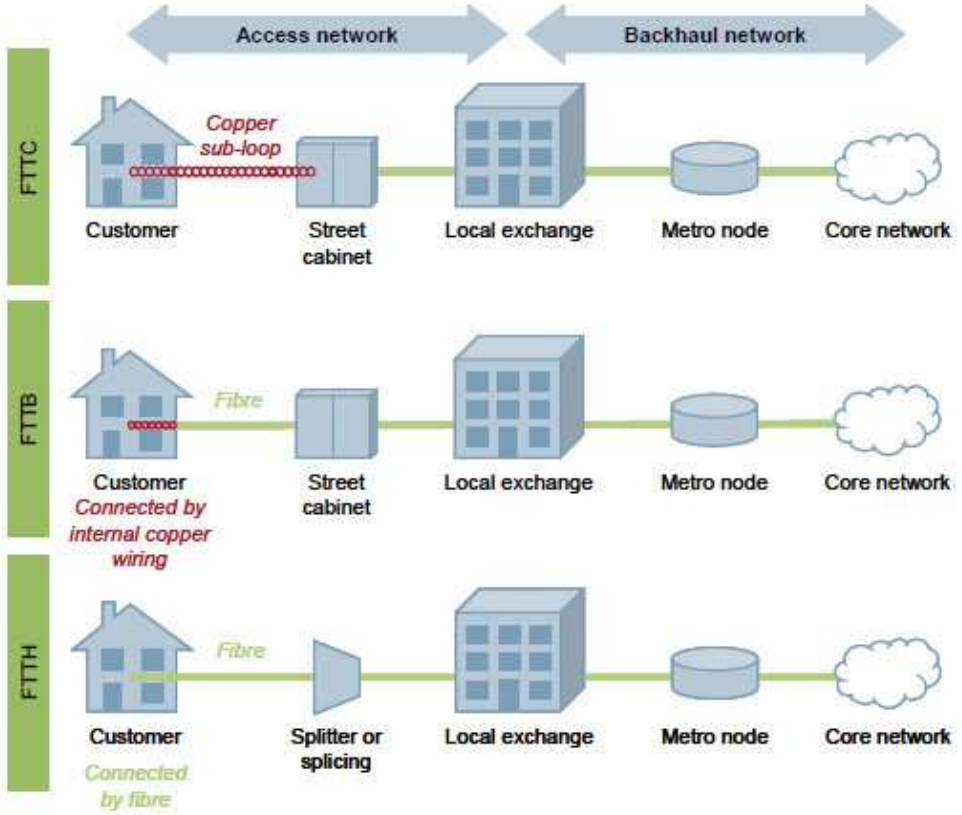
Metrics for broadband networks

Demand-side metrics

- surveys from regulators and other organizations, crowdsourcing apps
- penetration or adoption of broadband services
- usage patterns of applications and websites indexed by subscriber profiles
- statistics on security incidents, pricing and performance
- data from CSPs: connectivity, traffic volumes, QoS, usage patterns

Supply-side metrics

- broadband capacity, availability/coverage, and access speed
- indexed by technology (e.g. 3G/4G, FTTx) and providers



Access speed

Actual (measured) speed

- measured during tests, delivered in day to day use

Advertised (headline) speed

- typically peak (theoretical maximum), sometimes average value

Main considerations

- actual and advertised speeds can be vastly different
- speed depends on region, technology, sharing connections, time of day, application, protocols used etc.
- end-to-end vs. first-router speeds, download vs. upload speeds
- TCP: multiple parallel connections, many small requests (WWW) vs. single large request (video)
- selecting (sampling population of) subscribers within a region
- traffic shaping policies over the day (e.g. fair use policy)
- consistency in reporting to compare broadband by different CSPs

User defined metrics

Service penetration rate (SPR)

- fraction of service users among all users in given time period

Busy hour service attempt (BHSA)

- fraction of service users during busy hours in given time period

Concentration factor of service attempt (CSA)

- concentration of service uses throughout the day

Monthly service activity (MSA)

- concentration of service uses within days over a month

Service holding time (SHT)

- average time duration of each service use

Service throughput per usage (STPU)

- average traffic volume generated during each service use

Time interval of service attempts (TISA)

- average time interval between two consecutive service attempts by a user

Net data rate (NDR)

- average data rate of a service measured at the application layer

STANDARDIZED METRICS[1]

IP packet transfer delay (IPTD)

- time difference between ingress and egress packet events when such packet successfully delivered without errors

IP packet delay variations (IPDV)

- affected by TCP retransmissions, may cause buffer overflow or underflow

IP packet loss ratio (IPLR)

- fraction of lost packets

IP packet error rate (IPER)

- fraction of erroneously received packets

IP packet reordered ratio (IPRR)

- fraction of reordered but otherwise successfully received packets

Spurious IP packet ratio (SIPR)

- number of spurious packets observed during specified time interval

[1] ITU-T Y.1540 standard to measure QoS over the end-to-end heterogeneous connections

STANDARDIZED METRICS[1] (2)

IP packet severe loss block ratio (IPSLBR)

- fraction of the severe loss block outcomes

IP packet duplicate ratio (IPDR)

- ratio of duplicated packets to successfully received packets excluding duplicated packets

Replicated IP packet ratio (RIPR)

- ratio of replicated packets to successfully received packets excluding replicated packets

Service availability

- is IPLR¡0.75 for at least 5 min duration

[1] ITU-T Y.1540 standard to measure QoS over the end-to-end heterogeneous connections

STANDARDIZED METRICS[1]

Link/path bandwidth capacity (RFC5136)

- overall link/path bandwidth capacity

Bulk transport capacity (RFC3138)

- bandwidth capacity at the transport layer

One-way and two-way packet losses or connectivity (RFC2680)

- the number of packets lost

Packet one-way and two-way delay (RFC2679, RFC2681)

- end-to-end packet delay

Delay variation (RFC3393)

Packet reordering (RFC4737)

Duplicated packets

[1] IETF metrics for end-to-end QoS reflecting TCP retransmission mechanisms

Other broadband metrics

Operator oriented metrics

- ITU-T and IETF metrics just presented

Subscriber oriented (QoS) metrics

- upload and download speed
- round-trip time (RTT) delay/latency
- delay jitter
- packet loss
- DNS failure rate (fraction of failed DNS requests)
- DNS resolution (delay to translate URL to IP address)
- web browsing speed (average time to fetch complete website)
- average daily disconnection (number of interruptions a day longer than 30 s)
- distance from the digital exchange

Metrics by broadband regulators

Ofcom (UK)

- probability of download/upload speed greater than 2 Mbit/s
- probability of web-browsing loading speed below 1 s
- probability of latency less than 0.1 s

FCC (USA)

- download/upload speed
- web browsing performance with TCP
- UDP latency (average RTT of UDP packets)
- UDP packet loss
- video streaming test (initial time to buffer, buffer under-runs, delays)
- VoIP test (upstream/downstream packet loss and jitter, and RTT latency)
- DNS resolution (i.e. DNS delay)
- DNS failures
- ICMP latency (RTT of ICMP packets)
- ICMP packet loss
- latency under load (average RTT for UDP packets for varying loads)
- consumption (total bytes downloaded/uploaded by the router)

Internet measurements

Some considerations

- protocols: TCP vs. UDP, IPv4 vs. IPv6
- statistics: outliers, averaging
- conditions: avoid congestion, account for speed enhancing methods
- objectives: real-time monitoring vs. long-term capacity planning
- sampling: subscribers, packets, routers
- quality: accuracy and completeness

Tasks

- optimize the network resources
- identify anomalies and security issues
- establish traffic control policies
- set correct levels of service pricing
- classify traffic
- extract statistics of interest

Internet measurements (2)

Challenges

- measurements at or across traffic flows, or individual packets
- full, partial or no awareness of underlying protocol(s)
- tracking protocol state significantly improves traffic identification accuracy
- data often routed through different paths
- multiple simultaneous end-to-end connections and multiple sessions
- protocol encapsulation and tunnelling (e.g. IPv6 via IPv4)
- traffic encryption (e.g. HTTPS)
- proxies and caching

Deep packet inspection (DPI): Internet traffic attributes

- service tier, content provider
- operating system, browser, website, IP addresses, MAC addresses
- client device, client device type
- media stream type, application/session/transport protocols
- video/audio codec, media container, video resolution, OTT applications
- control versus data content

Common metrics for telecommunication networks

Metrics for telecommunication networks

Metrics sources

- technical literature, some used regularly (i.e. informal standards)
- industrial standards

Fairness

- either fair sharing of network resources among users
- or a single user is provided a fair access to network resources

Jain's index of fairness

- independent of network size and of the measure how resources are used

Max-min fairness

- resource utilization of some user cannot be increased if that user has already larger utilization of resources than other users

Proportional fairness

- useful in scenarios with utilization of multiple resources

Energy and power metrics

Main issues

- drivers: reduce the operational costs, and increase battery life-time
- measurements: location, duration, load, QoS constraints, applications used

Total energy consumed

- sum of operational and embodied energy

Operational energy

- energy consumed during the operation accounts for the RF power and overhead power

Embodies energy

- total energy over the whole life-cycle of equipment (manufacturing, transport, installation, decommissioning and disposal)

Energy consumption rating (ECR)

- ratio of expended power and maximum data throughput

Variable-load ECR

- ratio of weighted averages of power and data rates at several network loads

Energy and power metrics (2)

Energy efficiency rate (EER)

- inverse value of ECR

ECR for radio access networks (ECR-RAN)

- ratio of total expended power in the cell to the cell surface area

Power ratio of equipment

- ratio of output power to input power

ATIS energy metrics

- ratios of logarithm of expended power to capacity or throughput

ITU metrics

- ratios of expended power and product of throughput and distance (wireline networks), or throughput and area (wireless networks)

Key power indicator

- ratio of coverage (rural areas) or number of subscribers (urban areas) to total cell site power

Transceiver energy consumption

- given by powers in transmitting, receiving, scanning, idle, and sleep modes

Quality of service (QoS)

Main issues

- priced in service level agreements (SLAs)
- different performance indicators may target the same QoS
- QoS guarantees difficult due to dynamic mechanisms used (routing, channel allocation, energy saving, sharing resources etc.)
- QoS management: traffic flows, traffic prioritization and engineering

Application-oriented QoS (AQoS)

- concerned with end-to-end flows and can be used to assess user satisfaction

Network-oriented QoS (NQoS)

- concerned with optimizing core network of routers and switches
- measure capability to deliver services while efficiently utilizing resources

Network-oriented QoS (NQoS)

Throughput

- usually for a single flow, and is expressed in bits/s
- for single hop, end-to-end connection or aggregated for whole network

Computing or service throughput

- ability of system to process requests and to deliver the work in given amount of time

Packet delivery ratio (PDR)

- ratio of successfully delivered packets to total number of generated packets
- end-to-end packet losses usually inferred indirectly

Packet latency

- average end-to-end delay

Delay jitter

- variance of end-to-end packet delays in one or both (RTT) directions
- alternatively, difference between maximum and minimum RTT values
- jitter can be classified as random/deterministic, correlated/uncorrelated, and constant/transient/short-term

Availability

Main issues

- network should resilient to failures and provide agreed services most of time
- CSP standard is 99.999% (“five nines”) of service uptime
- unknown reliability of components and their interactions
- SLAs specify acceptable down-time and outage periods within given time
- can be assumed end-to-end, or at network level

Mean time to failure (MTTF)

- expected time to next failure

Mean time to repair (MTTR)

- average time to return to operational state after a failure

Mean time between failures (MTBF)

- average time between two successive failures

Impacted user minutes (IUM)

- product of number of users affected by failure and failure duration

Defects per million (DPM)

AVAILABILITY (2)

Point availability

- probability the system is operational at future time given last repair time

Average uptime availability

- proportion of time the system is ready to deliver service

Steady state availability

- long-term probability for given rates of failure and repair

Inherent availability

- steady-state availability considering only corrective downtime

Achieved availability

- availability assuming only the planned shutdowns

Operational availability

- average availability assuming all expected downtimes

Quality of experience (QoE)

Main issues

- unknown non-linear human senses processing, lack of appropriate models
- fast pace of technology development
- shift from network-centric to user-centric design

QoS vs. QoE

- complex relationship, aim is to infer QoE from QoS
- improving QoS does not guarantee better QoE

Subjective QoE metrics

- take the human perspective on perception of quality differences
- mainly based on measured statistics using user surveys (slow and laborious)
- not considered by standards nor for good for real-time applications
- however, used by CSPs to accurately forecast consumer satisfaction

Subjective QoE metrics (2)

Mean opinion score (MOS)

- numerical QoE index evaluated through subjective tests

Double stimulus continuous quality scale (DSCQS)

- index of video quality, less sensitive to context, not for real-time evaluation

Single stimulus continuous quality evaluation (SSCQE)

- quality monitoring of real-time applications

Absolute category rating (aka single stimulus)

- efficient, reliable, standardized method permitting different test conditions

Double stimulus impairment scale (DSIS)

- paired evaluation of reference video against the impaired video

Single stimulus continuous quality evaluation (SSCQE)

- uses a slider device and no standard video

Just noticeable difference (JND)

- a scale obtained by series of comparison tests on two samples

Maximum likelihood difference scaling (MLDS)

- relative difference in quality to represent utility of tested parameter on quality

Objective QoE metrics

- utilize algorithms, data and mathematical models to infer user satisfaction
- data are often QoS measurements, can be used adaptively and in real-time
- use reference signal (image, video): full-reference (FR) reduced-reference (RR), and no-reference (NR) metrics
- main challenges: unknown dependence on system parameters, user satisfaction is time varying, so needs to be tracked

E-model

- real-time estimation of mean opinion score (MOS)

Perception evaluation of speech quality (PESQ)

- estimates MOS by comparing observed signal with a reference

Application performance index (APDEX)

- satisfaction on scale between 0 (no users satisfied) to 1 (all users satisfied)

Peak signal-to-noise ratio (PSNR)

- inversely proportional to mean square error (MSE)

Moving picture quality metric (MPQM)

- video quality index using content dependent factors and network impairments (e.g. packet losses and delays)

Objective QoE metrics (2)

Motion-based video integrity evaluation (MOVIE)

- evaluates video impairments jointly in space and time

Structural similarity index (SSIM)

- degradation of structural information as luminance and contrast

Video quality metric (VQM)

- detects human perceivable artifacts for given codec type, block and color distortions

Pseudo subjective quality assessment (PSQA)

- real-time evaluation of quality of video/audio communications over packet networks

User satisfaction index (USI)

- exploits rigorous analysis of network level QoS during the video/audio call

Security metrics

Main issues

- ill defined, lack of sufficiently accurate security models
- cannot be inferred from other parameters (e.g. as QoE from QoS)
- industry defined security metrics and frameworks just emerging
- security incident detection/protection is different from security metrics
- involvement of human factors makes the security very complex

Defining security metrics

- a lot easier to indicate relative perceived security difference rather than providing the absolute measures
- hence, useful to consider a baseline system to evaluate effectiveness of security policies and compare security mechanisms

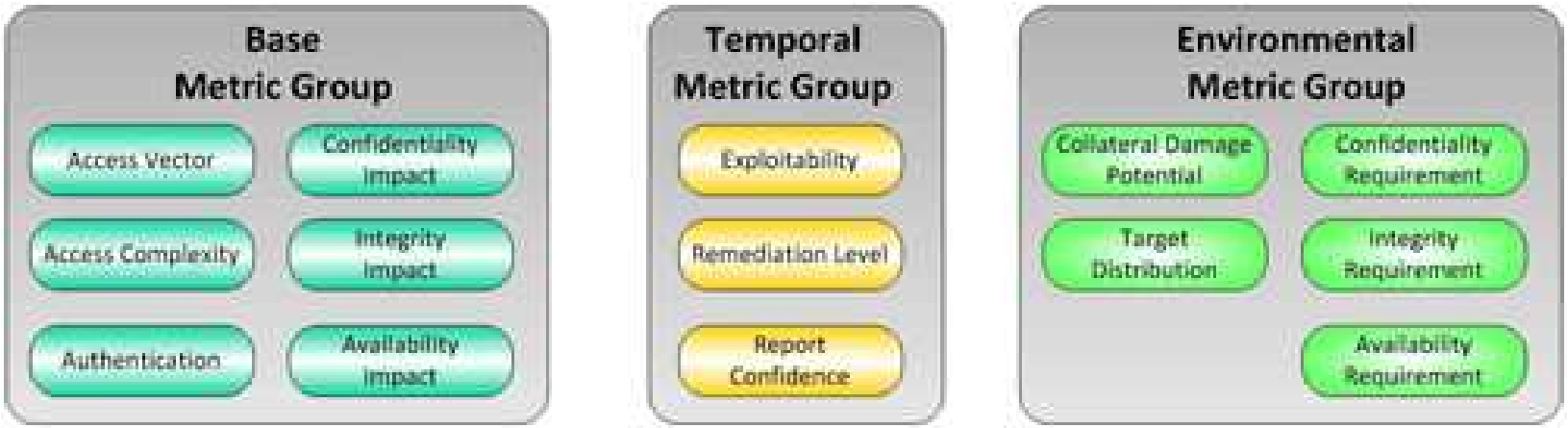
Attack graphs

- visualization of attack progression through system vulnerabilities
- more vulnerabilities, more attack possibilities
- usually generated for a network host which needs to be protected

SECURITY METRICS (2)

Common vulnerability scoring system (CVSS)

- universal language to describe system vulnerabilities and their urgency to prioritize the response and defences
- industry standard on defining and accessing the security of systems
- does not score threats, real-time attacks nor manage security risks



- vendors computed: base scores for severity, temporal scores for urgency
- user computed: environmental scores
- all scores combined to obtain a single security score for the system
- there are attempts to further optimize CVSS framework for specific systems

User defined security metrics

VEA-bility metric (vulnerability, exploitability, attackability)

- combines impact and temporal assessment of vulnerability using CVSS with exploitability, network topology and attack graph
- evaluated for each network host, then combined into one final value

Mean time-to-compromise (MTTC)

- average time required by attacker to compromise the network
- assumes attack graph and probabilities of security events

Relative cumulative risk (RCR)

- score evaluating host vulnerability and its proximity to the untrusted hosts

Hazard metric

- evaluates security risk by assessing maturity level, frequency of exploits, exploitability impacts, and level of authentication

Critical Vulnerability Analysis Scale Ratings (CVASR)

- uses questionnaires to collect security data and assess security threats

Weakest link security, Mean-Time-To-Problem-Report (MTTPR), Mean-Time-ToProblem-Correction (MTTPC), Problem exposure/correction/exploit rates

Robustness and resilience metrics

Failures

- occasional and random or targeted and large-scale
- cascading, due to overloading attacks and other
- structural related to interactions of components, or functional related to dysfunction of components

Robustness

- ability of a network to withstand failures i.e. fault-tolerance
- rich connectivity to reroute traffic when topology changes or congested links

Survivability and Quality of Recovery (QoR)

- ability to recover from failures and attacks within maximum outage duration

Resilience

- ability to provide acceptable level of service despite failures
- resilience is long-term unlike robustness is short-term focus

Key issue

- self-organization and autonomy increase complexity but also create vulnerabilities, so robustness and resilience are decreasing
- hence, robustness and resilience can be used to infer the level of security

Resilience metrics

Main issues

- network designed for QoS (and recently also for QoE), resilience secondary
- usually derived for static graph topology, however, largely unexplored dynamic and virtualized topology

Some metrics

Node connectivity	Average neighbor connectivity
Heterogeneity	Average node degree
Symmetry ratio	Clustering coefficient
Average hop-count	Radius
Closeness	Betweenness
Diameter	Average shortest path length
Algebraic connectivity	Natural connectivity
Weighted spectrum	Network criticality
Effective graph resistance	Path diversity
Assortativity coefficient	

- these metrics are defined in Network Science textbooks

Big data metrics

Data volume

- amount of data generated per a unit of time

Data velocity

- speed at which the data are being generated and moved around

Data variety

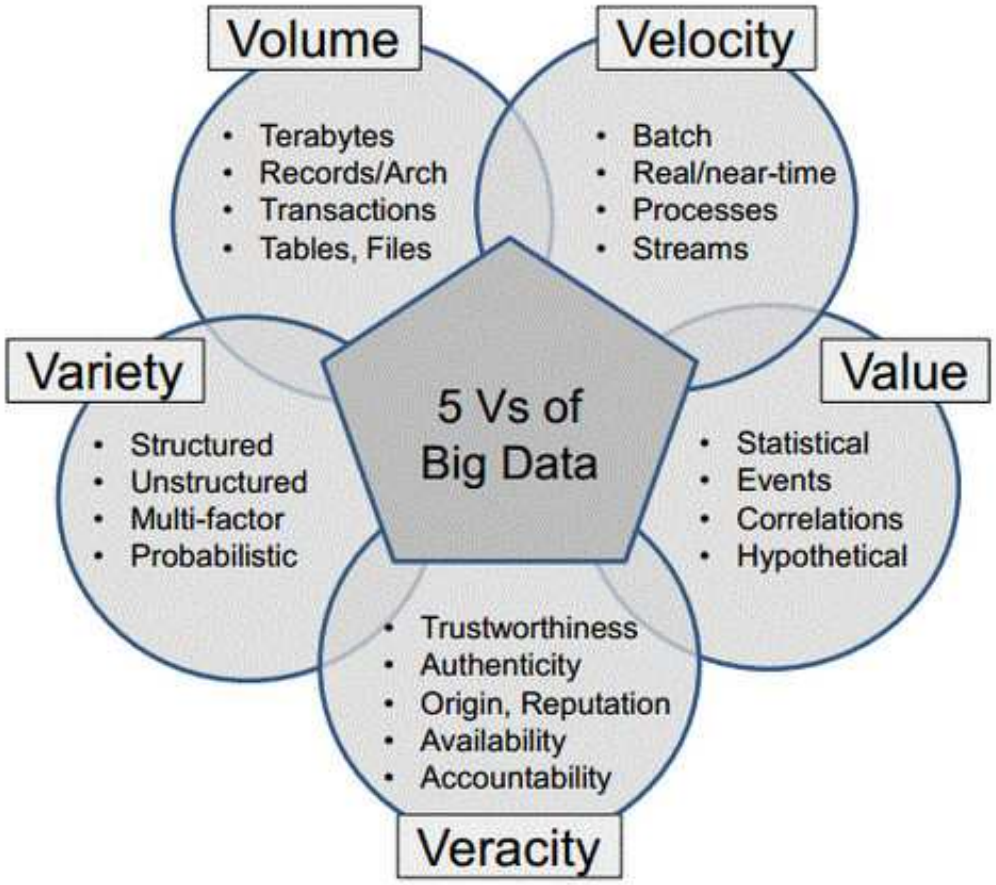
- refers to different sources and types of data

Data veracity

- level of trustworthiness of data

Data value

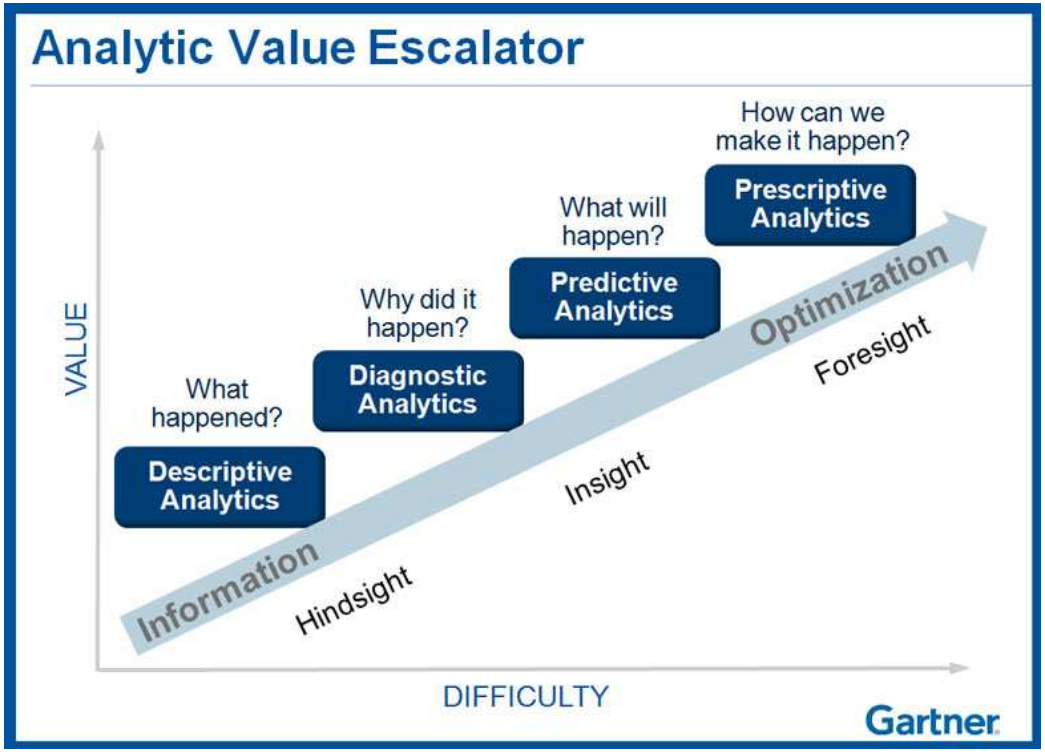
- potential monetary or other valuation of data



Machine learning (ML)

Main issues

- new advanced techniques: virtualization, softwarization, resources slicing
- pre-defined metrics not adequate in highly adaptive/autonomous systems
- some problems cannot be clearly defined e.g. security audit, fault prediction, revenue maximization, pricing adjustments etc.
- key objective: support decision making, enable big data analytics, but decisions not justified nor explained
  $\longrightarrow$ ML algorithms to validate other ML algorithms



ML PERFORMANCE METRICS[1]

Estimator metrics

- estimator quality: variance and bias, mean squared error

Classification metrics

- loss, scoring and utility functions
- e.g. accuracy score (proportion of correct identifications
- false-positives/negatives, sensitivity and specificity

Regression metrics

- e.g. mean absolute error, and r2 score

Clustering metrics

- similarity and distance metrics
- more generally, kernels are used for more complex objects such as graphs

[1] See documentation for Python *scikit* ML library at http://scikit-learn.org

Conclusions

Metrics and sensing

- became a lot more complex: 1000's of metrics to select and maintain for 1,000,000's of system parameters
  → any 2 metrics can represent some trade-off
- deluge of user defined metrics
- transition from a few metrics to metrics and sensing frameworks
- cannot separate the Internet backbone from supported physical systems

Metrics most actively studied in 2017

- QoE as CSPs move from QoS
- security and quantifying and enumerating vulnerabilities and risks
- using ML based metrics which are trustable

Thank you!