Beyond Benchmarking: Statistical Sound Experimentation

Kees van Hee

ECSS 2012, Barcelona

Technische Universiteit
Eindhoven
University of Technology

Where innovation starts

Outline

1. **Experimentation in CS, why and why not?**
2. **A new approach to benchmarking**
3. **Towards Experimental CS**

1. Experimentation why not?

- **CS was founded in competion by electrical engineers and mathematicians*: the last group won!**

  ***and a little by business administrators**

- **Mathematicians live in a world of models they create themselves**
- **Software engineers make a model of a *system* and use it as a specification for its construction**
- **So no experiments needed, but only formal proofs:**
  1. **The model is consistent and satisfies the requirements**
  2. **The system is a correct implementation of the model**
- **Today model checkers can deal with systems of a realistic size, theorem provers as well!**

Experimentation is needed…

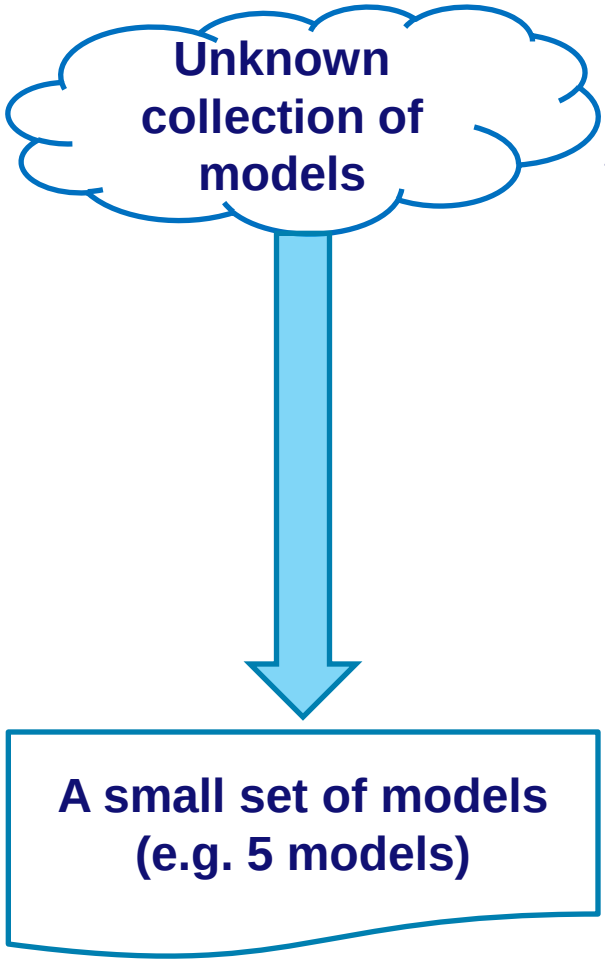
- **User requirements are always incomplete and only partly formal: we need experimental validation***
- **One modeling formalism is not enough (think of UML family), So we have to deal with different *aspect models*: formal verification of consistency is infeasible in practice**
- **In practice it is impossible to *verify*** if software is a correct implementation of a model: so *validation* by *testing* is necessary**

*validation is a *statistical* proof by an experiment
**verification is a *formal* proof

Experimentation is needed…

- **Software engineering processes are very difficult to compare:**
  - **No good models to compare them**
  - **Data acquisition to compare them is extremely expensive**
- **Quality of heuristic methods (like genetic algorithms) can only be established by experiments**
- ***Worst case* analysis of algorithms is often possible but *average* performance requires insight in the *distribution* of problem instances**
- **Often *bench marking* is used: a fixed sample of the distribution of instances**

2. New approach to benchmarking



Model collection:

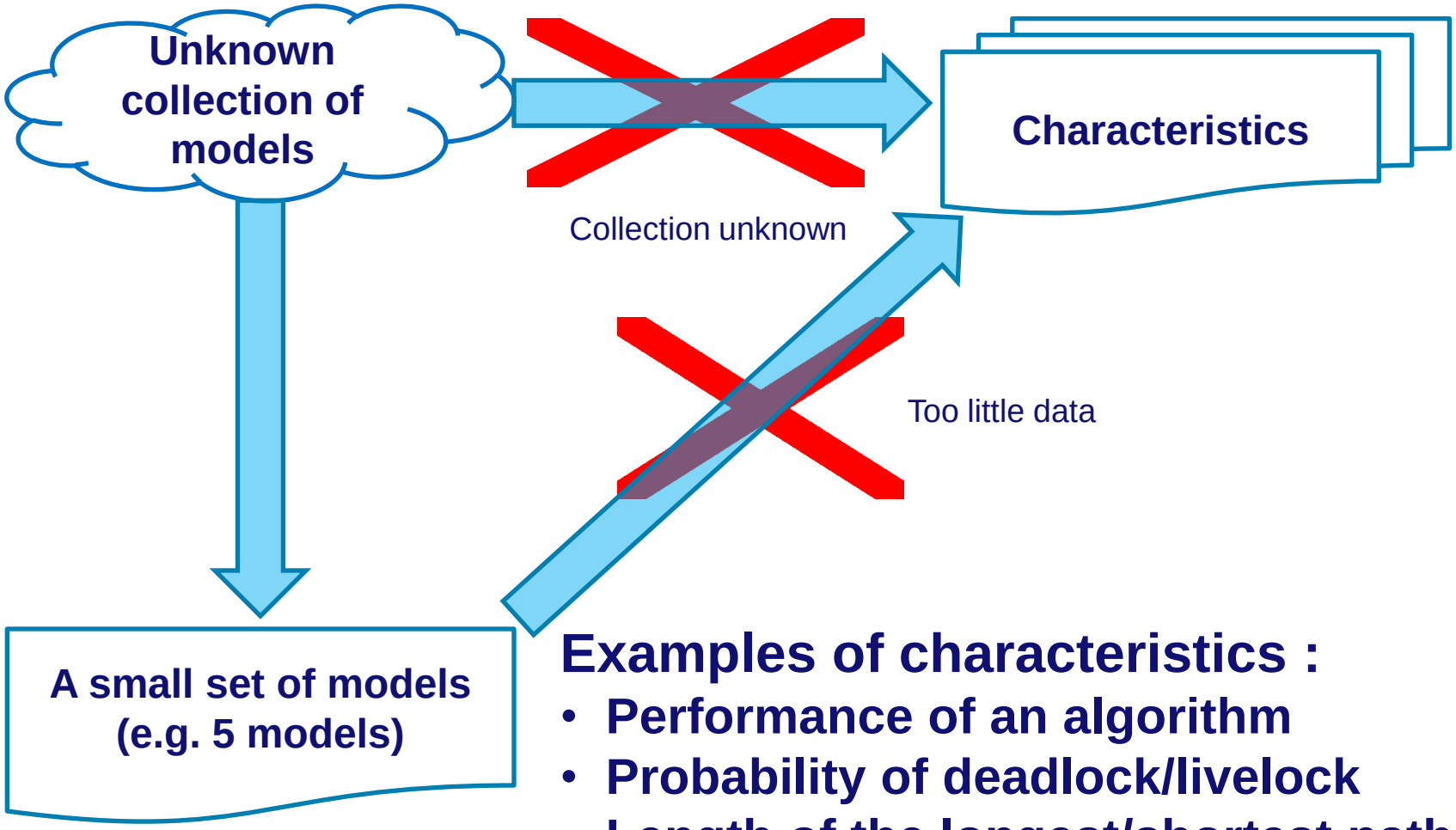
- **finite or an infinite set of models**
- **each with its own probability of occurrence**

Consider:

- **Model as *problem instance***
- **Model collection as *problem class***

Based on paper:
K. van Hee, M. La Rosa, N. Sidorova and Z. Liu
Discovering characteristics of stochastic collections of process models.
Business Process Management 9. LNCS 6896 (2011)

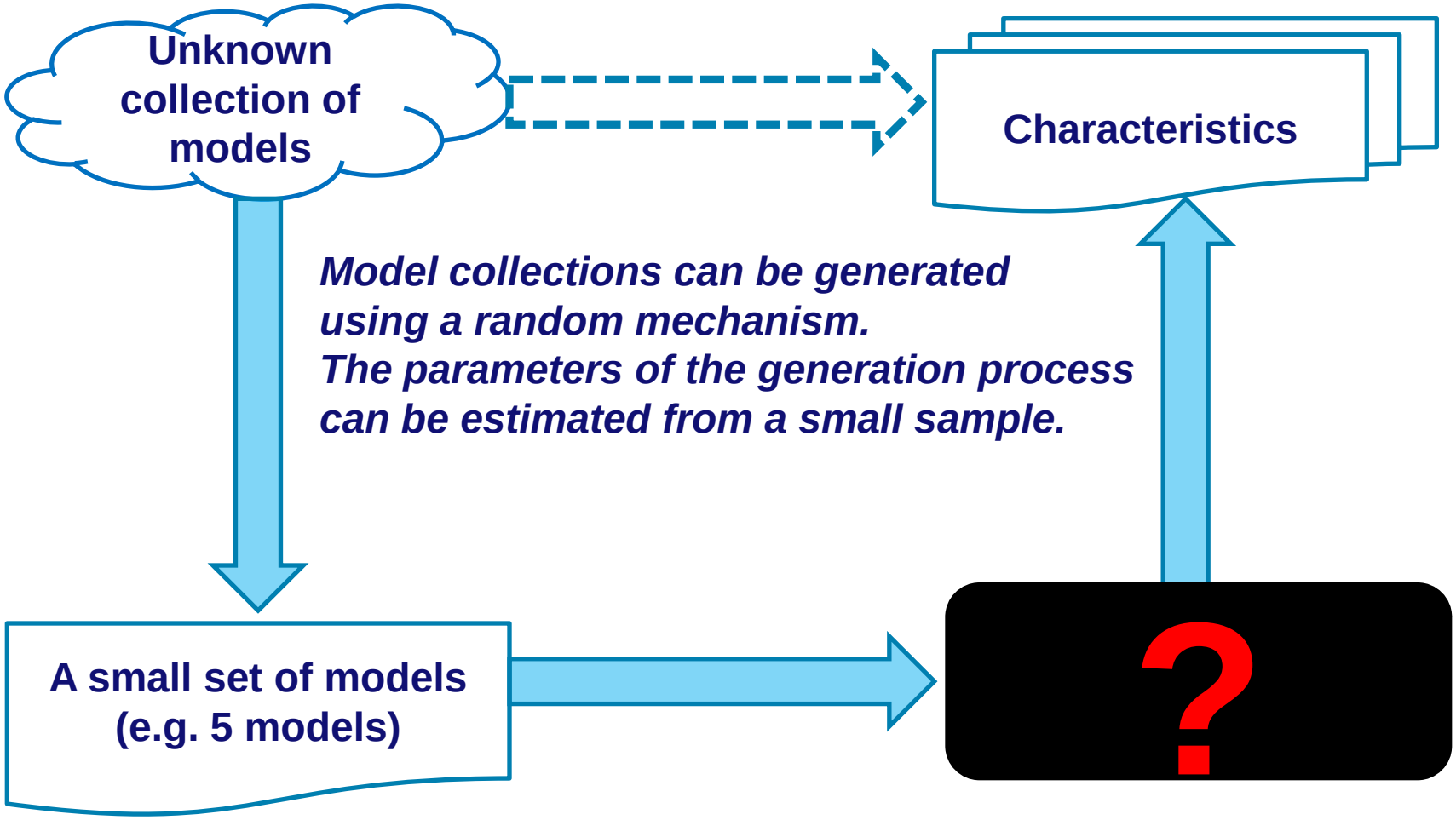
We want to know collection characteristics



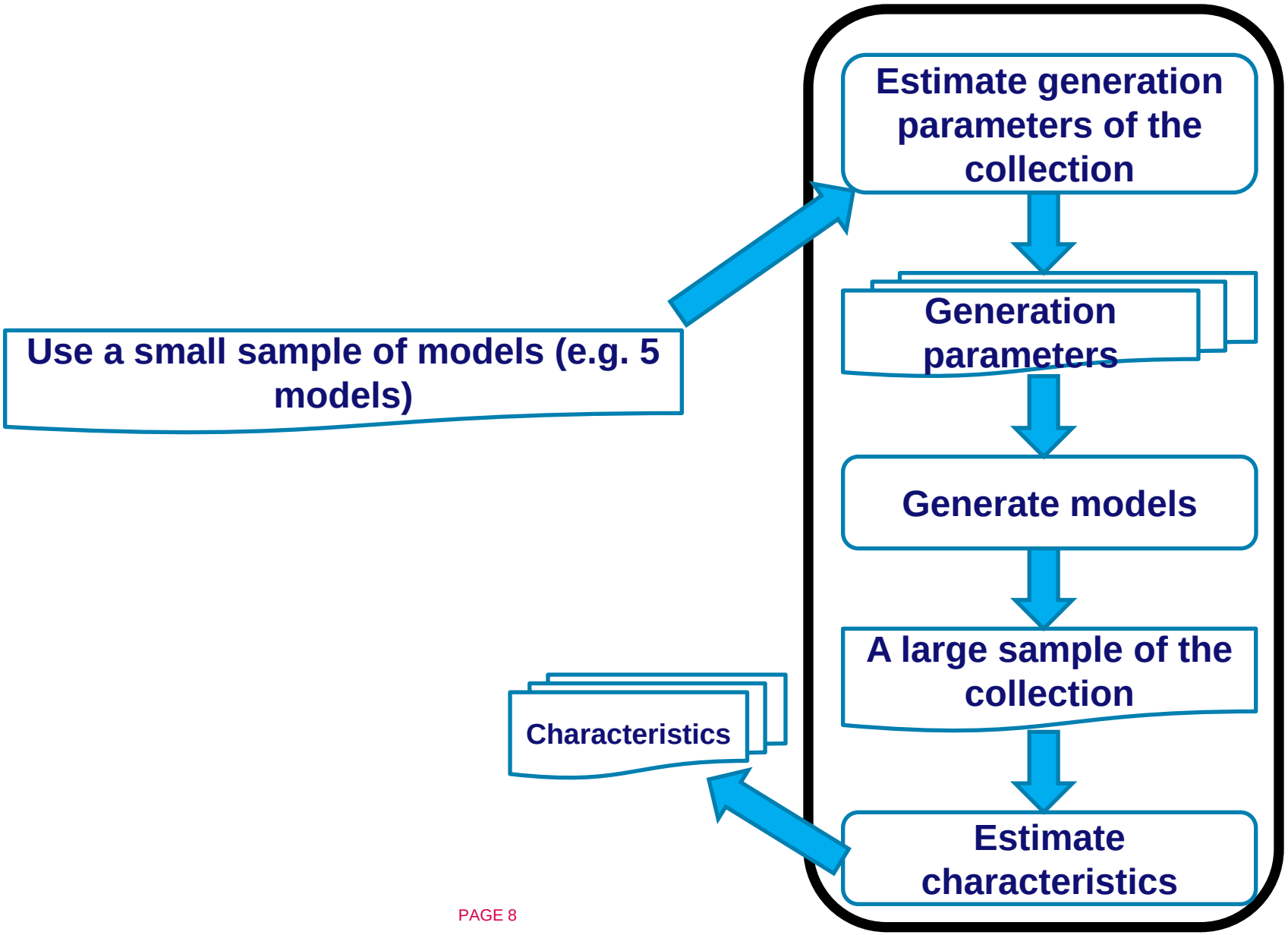
Examples of characteristics :

- **Performance of an algorithm**
- **Probability of deadlock/livelock**
- **Length of the longest/shortest path**

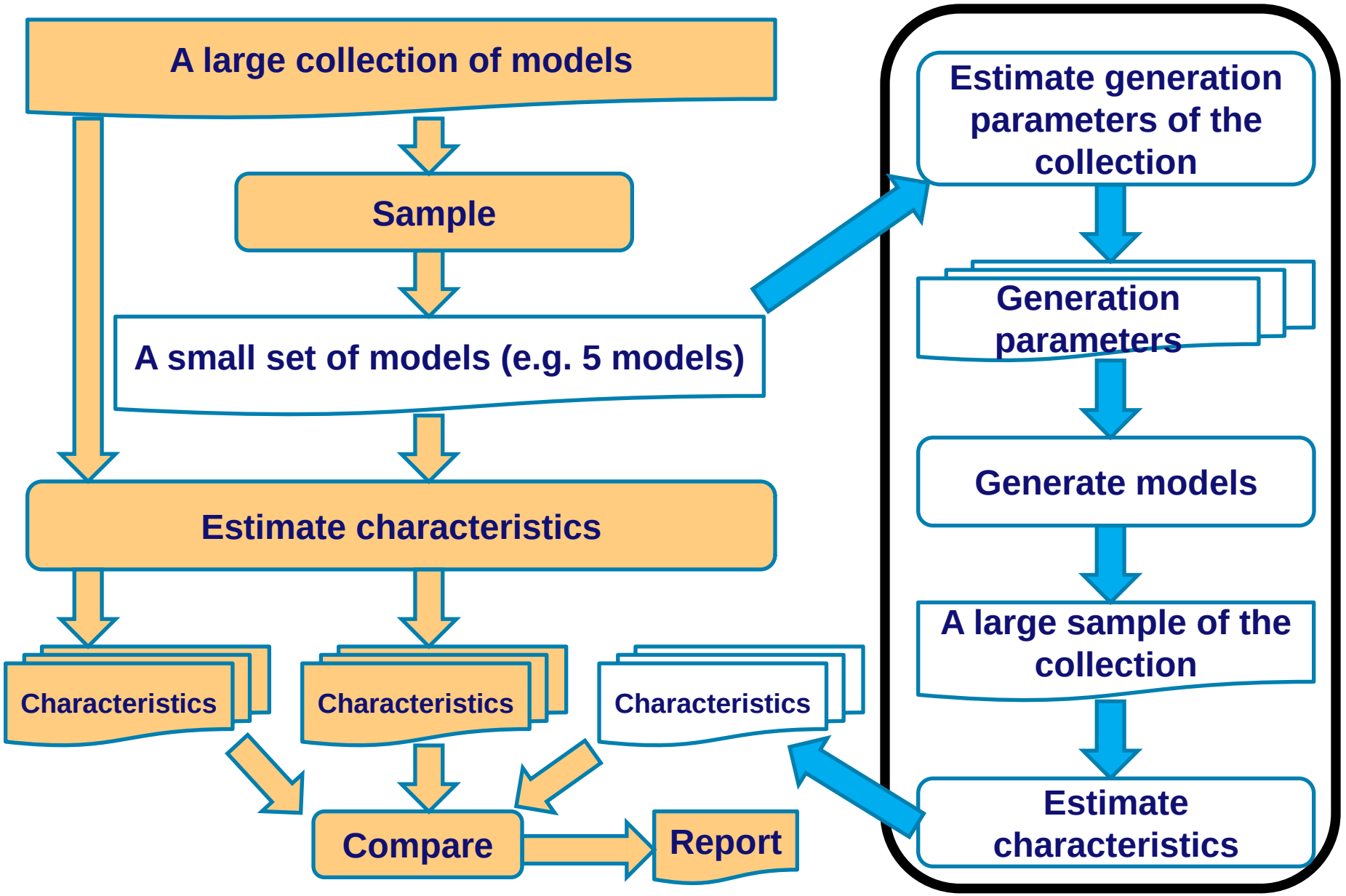
Assumption:


Unknown collection of models
Characteristics
Model collections can be generated using a random mechanism.
The parameters of the generation process can be estimated from a small sample.
A small set of models (e.g. 5 models)
?

Overview of the approach



To validate the approach…

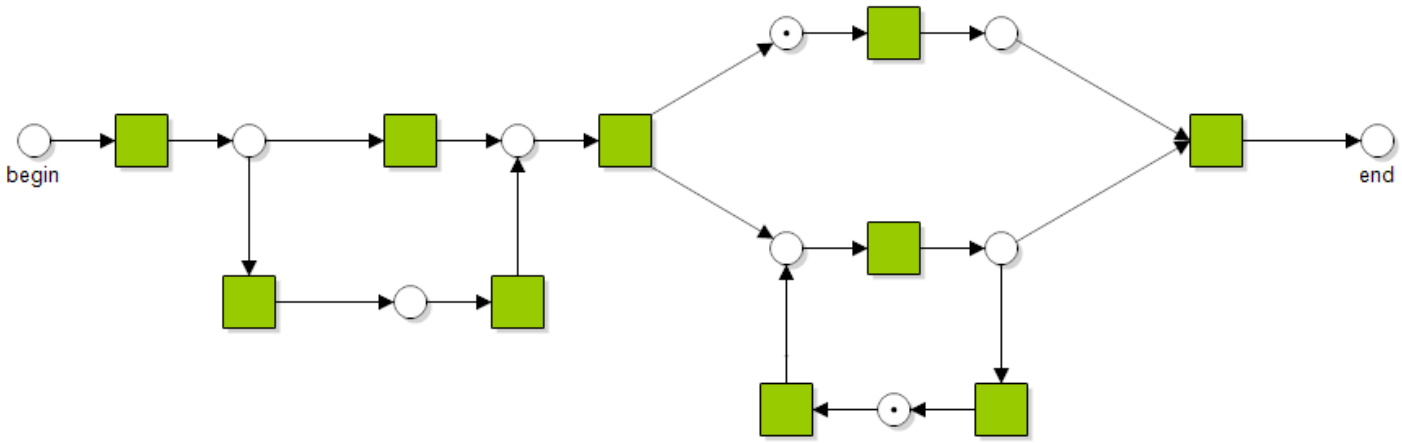


Example model collection: Petri nets

In particular:

***Workflow nets* are a special class of Petri nets:**
- **One input and one output place**
- **Every node on a directed path from input to output place**

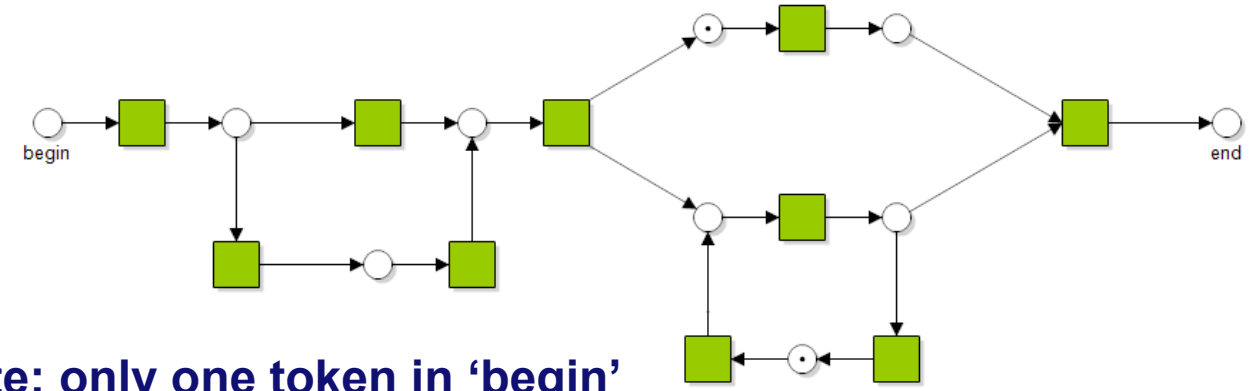
Workflow nets are used to model:
- **Parallel (multi threaded) programs**
- **Business procedures**



Example of characteristics

1. Mean length of *longest path* (without loops)
2. Mean length of *shortest path*
3. Probability of *weak termination**

***weak termination: from each state reachable form the initial state, the final state is reachable**

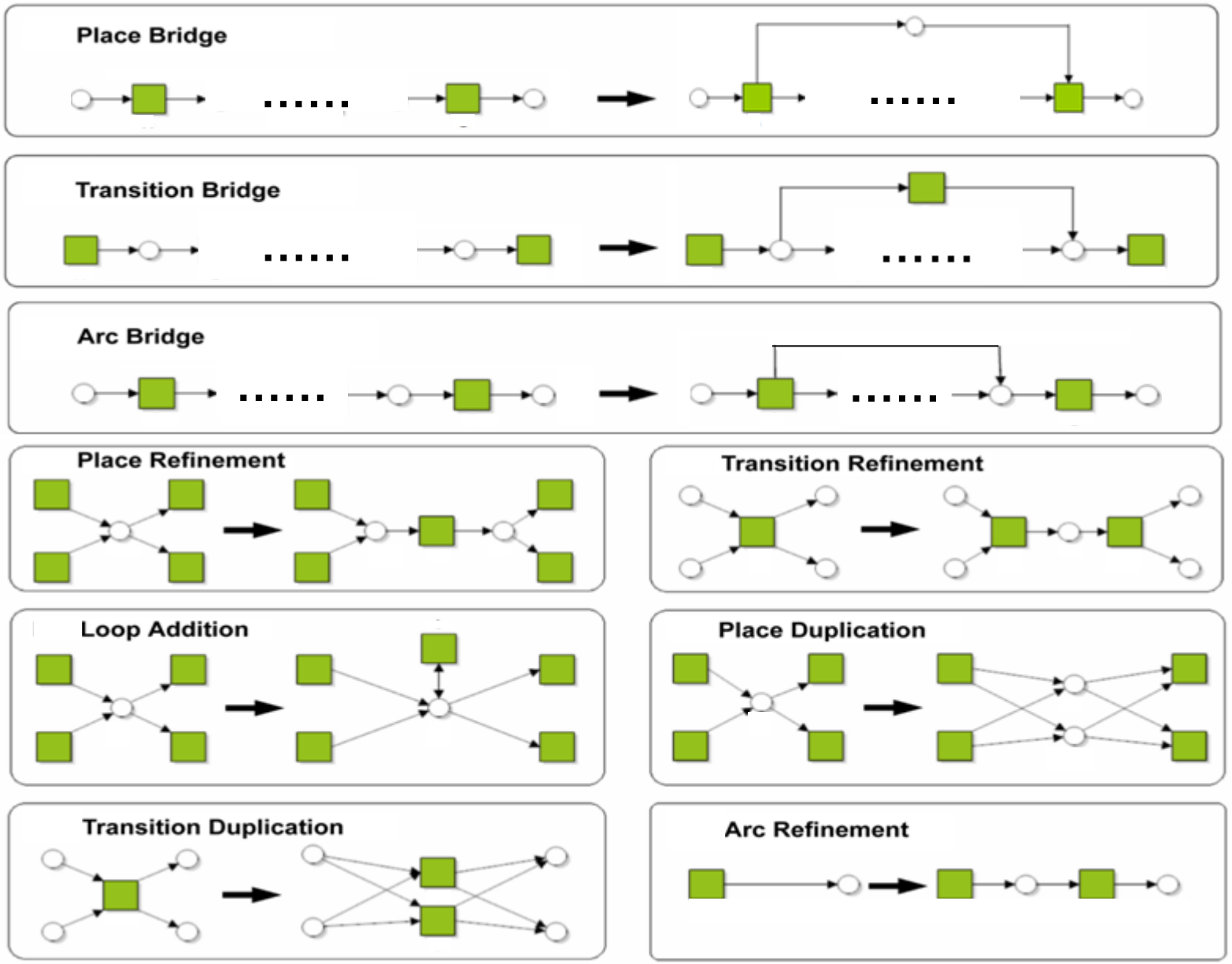


Initial state: only one token in ‘begin’
Final state: only one token in ‘end’

Generation of Petri nets

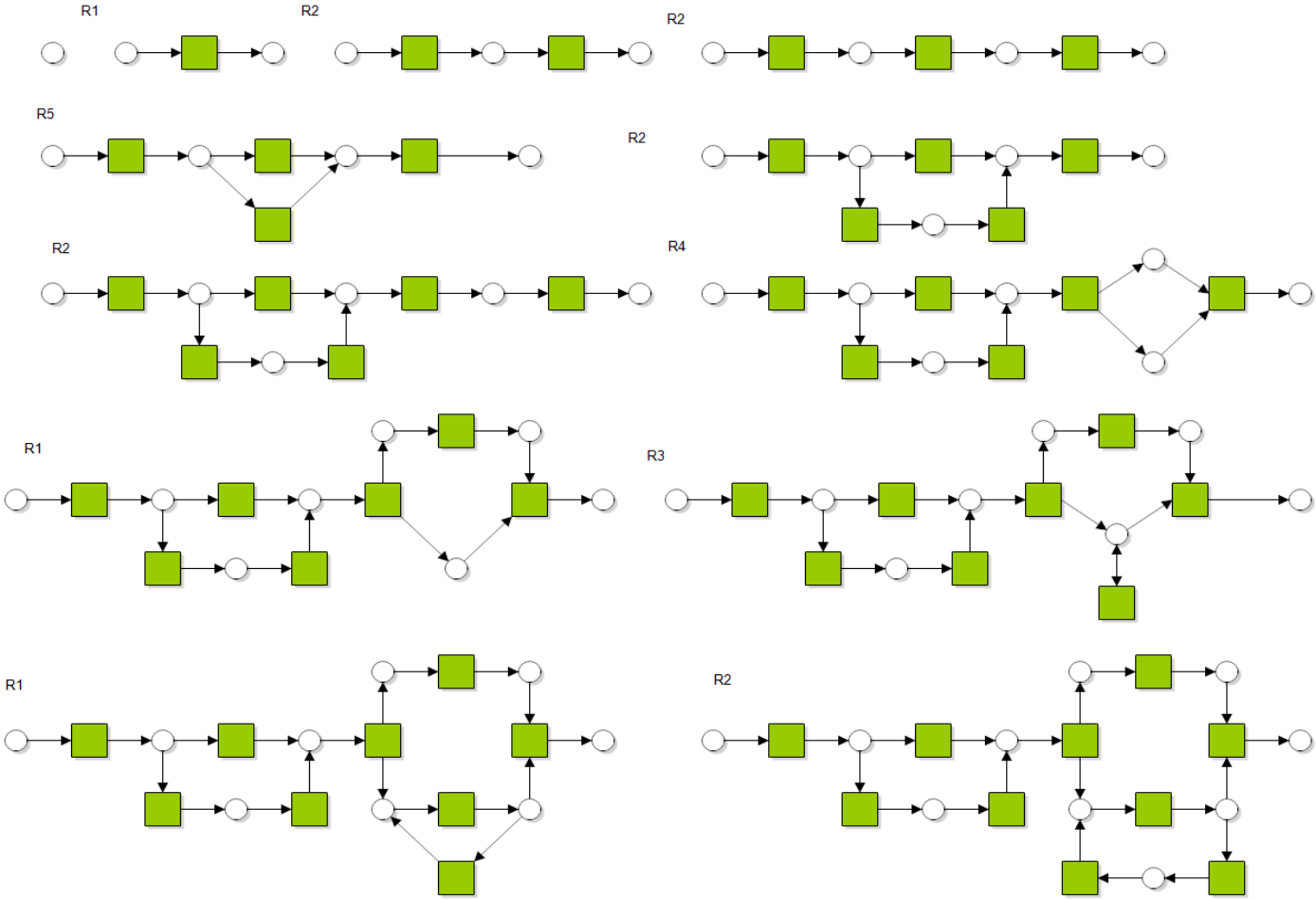
- **Generation uses a *stepwise refinement* approach**
- **Three determinants of workflow net generation**
  - **Construction rules**
  - **Probabilities of applying the rules**
  - **Number of construction steps, called *size***
- **We call them *generation parameters***
- **Any Petri net can be generated, starting from a set of places**

Construction rules

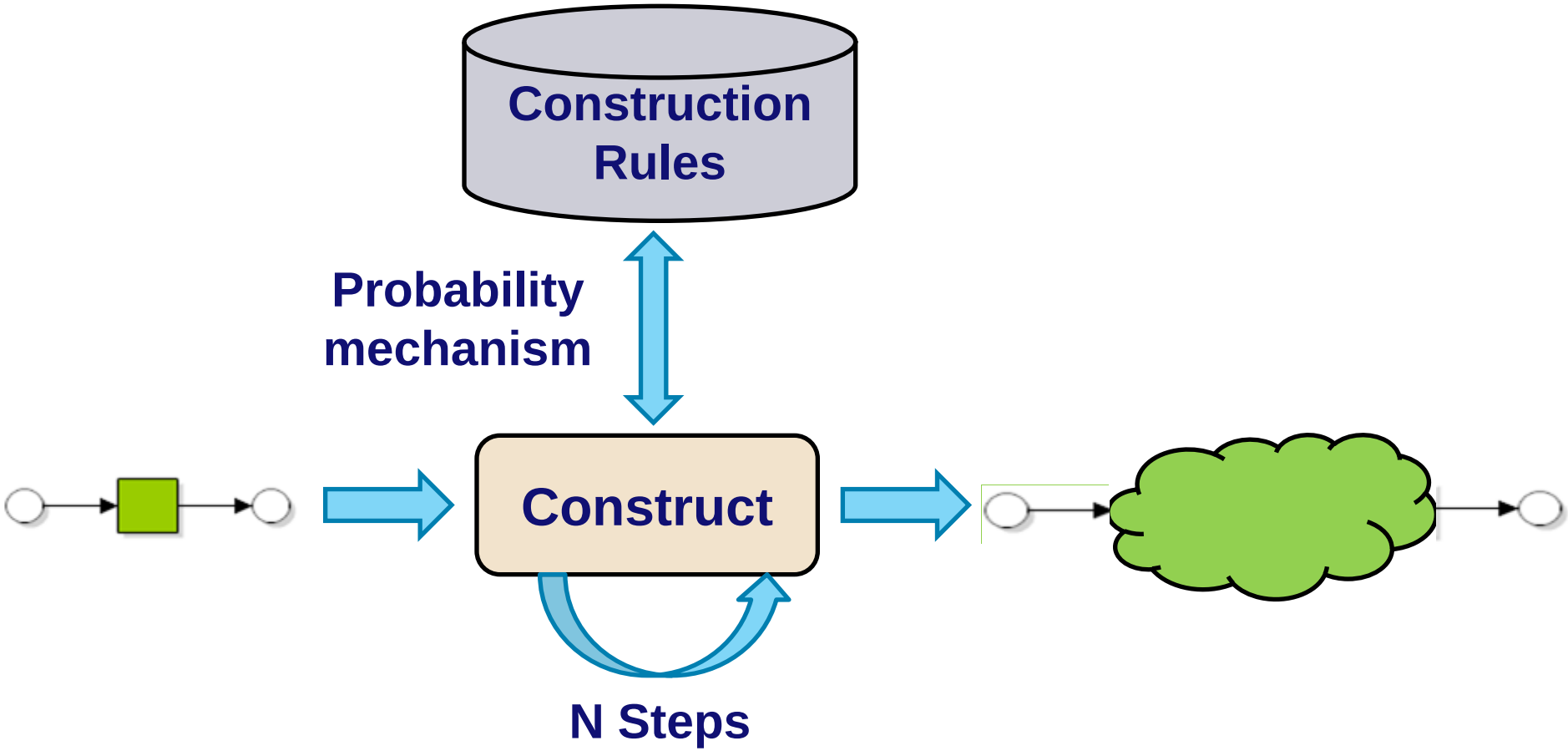


Construction of workflow nets

Starting with one place and applying rules R1…R5 only, gives weakly terminating wfnets!



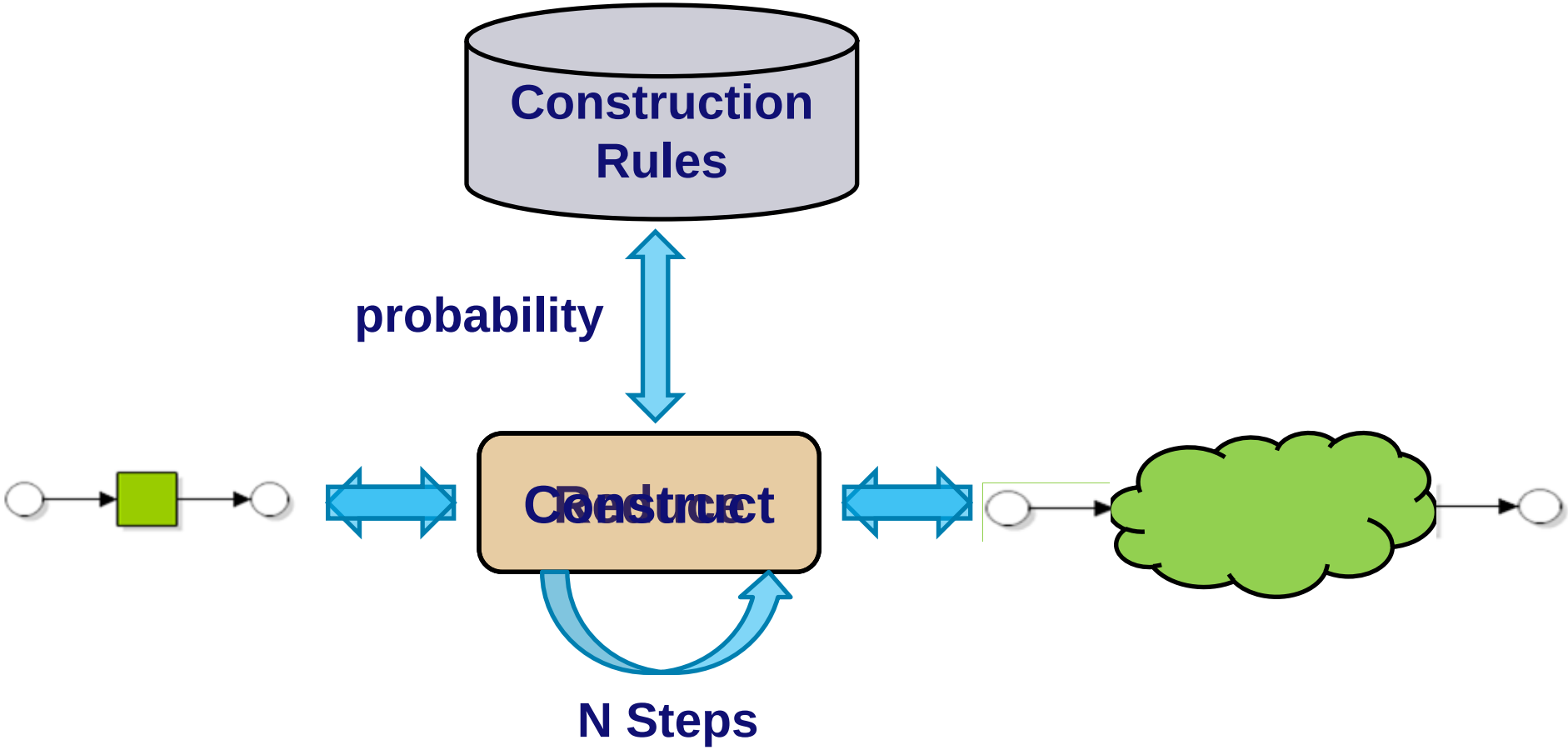
Procedure of generating a workflow net



Parameters of generation process

- $p_{n,r}$ is the probability that rule *r* is chosen to extend net *n*
- $q_{m,r,n}$ is the probability that net *n* is the result of applying rule *r* to net *m*:
- *s* is the distribution of size
  e.g. stopping is based on Poisson distribution

Estimation of generation parameters

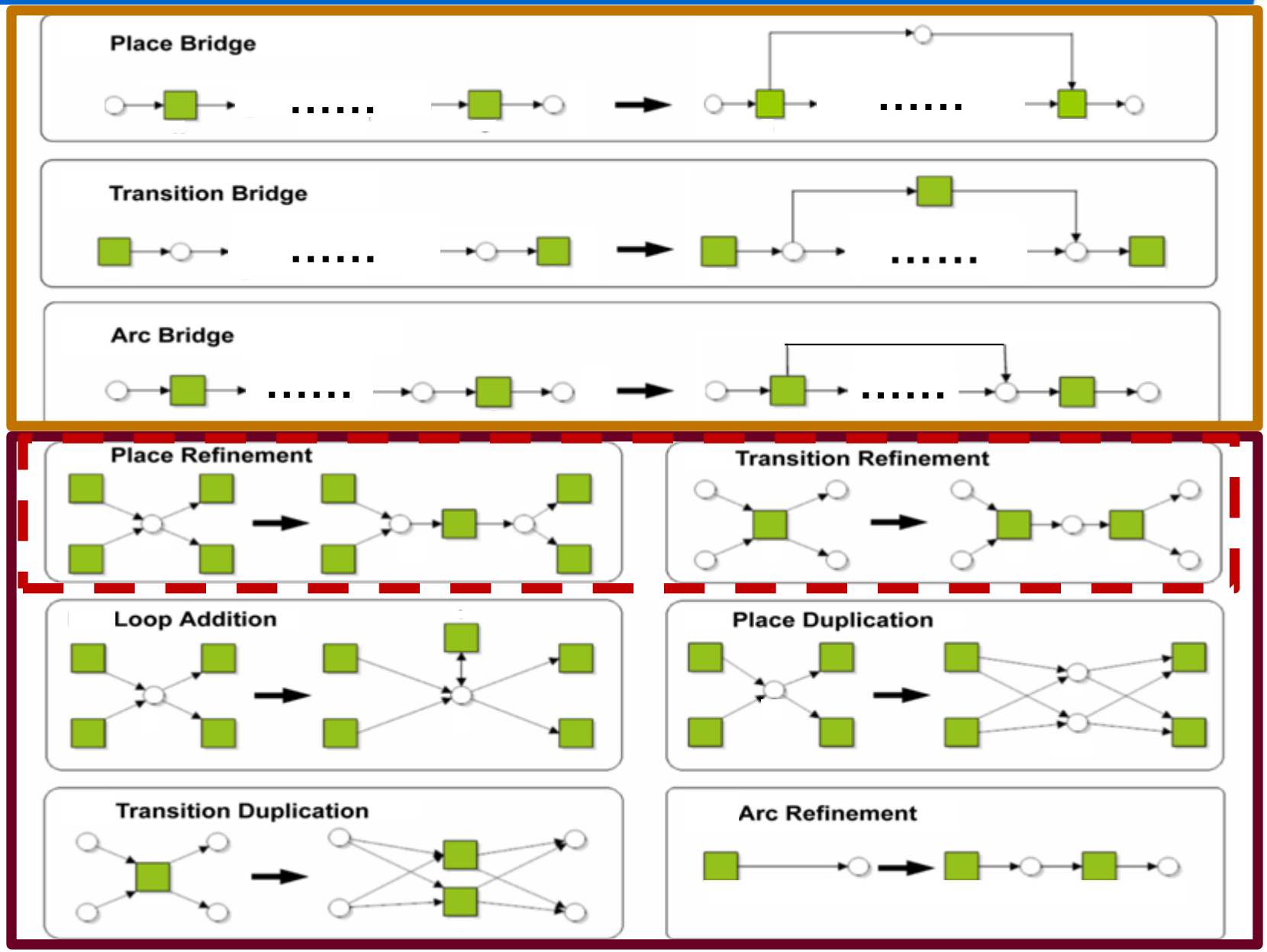


Count the reduction steps per type in each sample

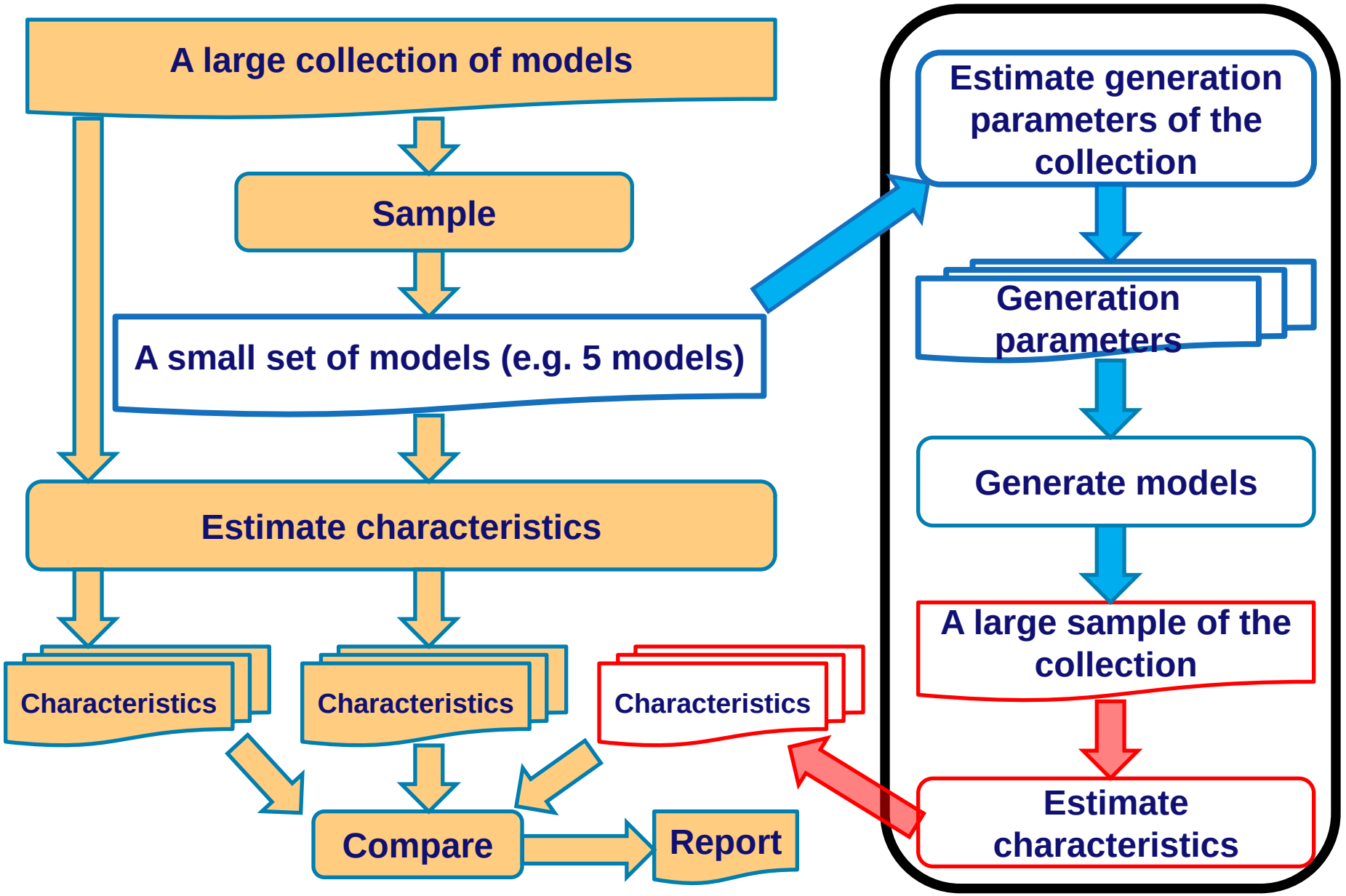
Priority of rules

2

1



Estimation of characteristics



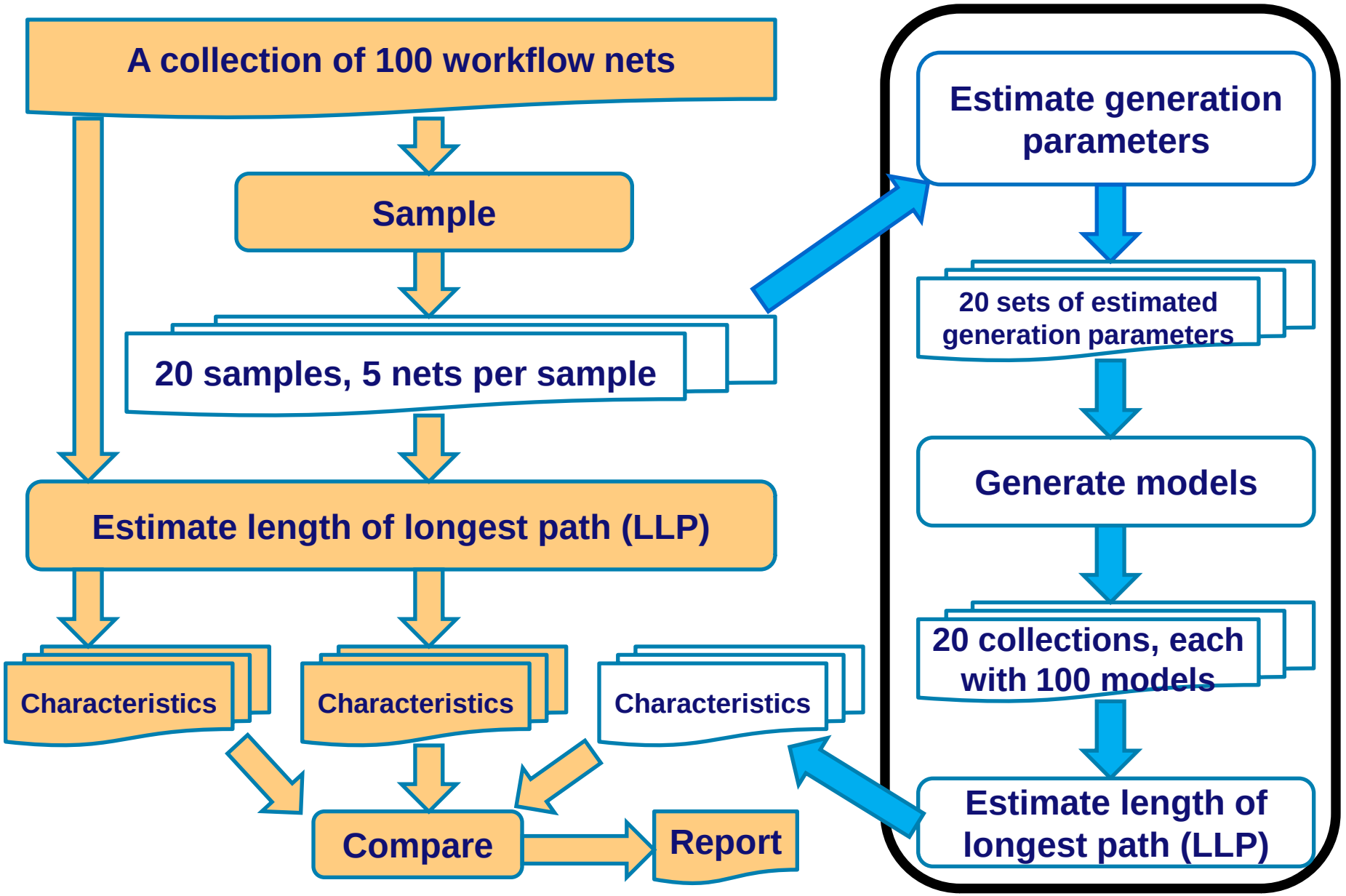
Estimation of characteristics

- **Generate an arbitrary large sample from a given small sample of the original collection**

Bootstrapping!!

- **Based on the law of large numbers and the central limit theorem, we can estimate the characteristics of the original collection with any precision**

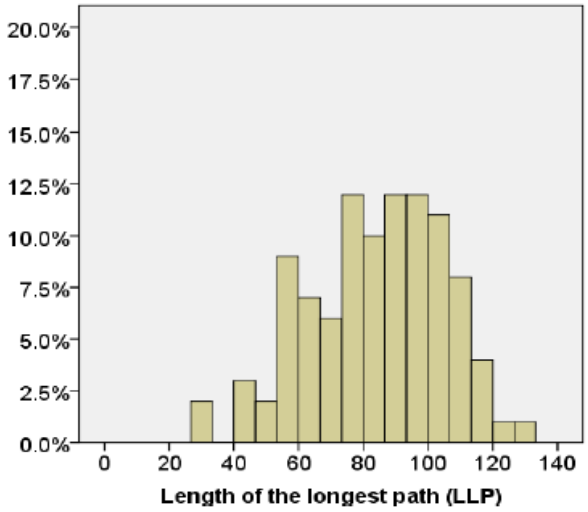
Experiments with artificial models



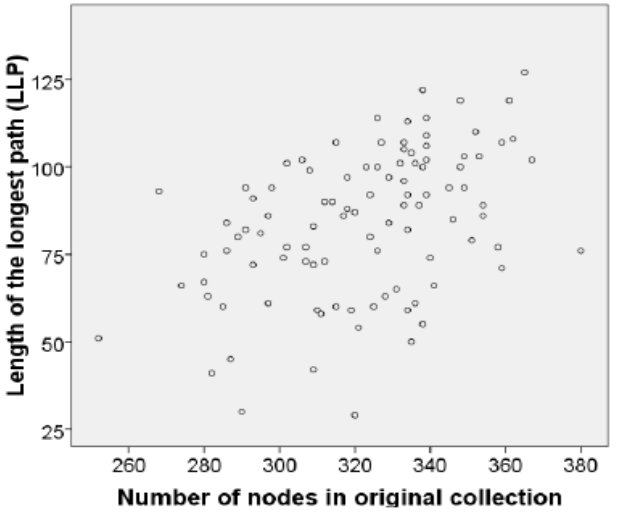
Result of generation parameters

	Probability of Refinement rules	Probability of Transition Dup.	Probability of Place Dup.	Size mean
original parameters	0.60	0.20	0.20	200
parameters' estimations 1	0.58	0.21	0.22	201
parameters' estimations 2	0.63	0.18	0.19	194
parameters' estimations 3	0.61	0.21	0.19	209
parameters' estimations 4	0.61	0.20	0.20	203
parameters' estimations 5	0.59	0.20	0.21	206
⋮	⋮	⋮	⋮	⋮
parameters' estimations 20	0.60	0.20	0.20	207
Avg.	0.60	0.20	0.20	201
Std.Dev.	0.0162	0.0168	0.0105	8.6601

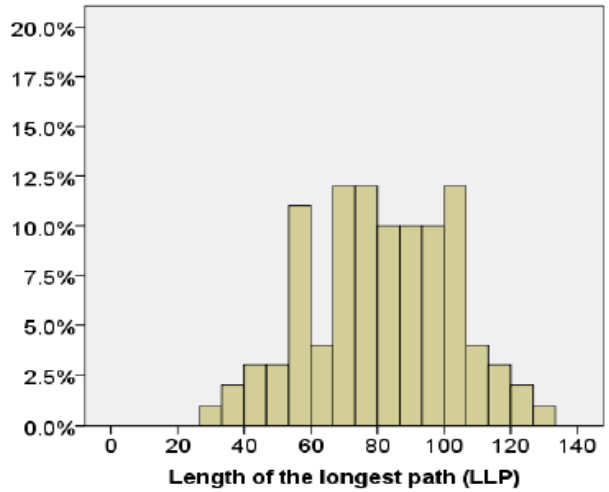
Result of distribution of length of the longest path



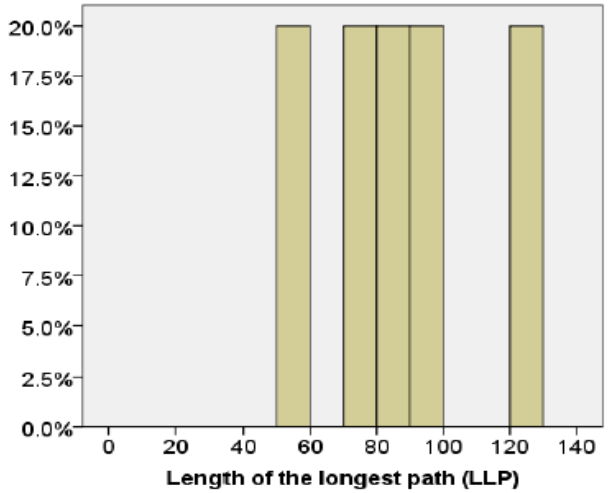
(a) Histogram of the LLP distribution in the *original* collection



(b) Scatter plot relating the number of nodes and LLP in the nets of the *original* collection



(c) Histogram of the LLP distribution in the *generated* collection

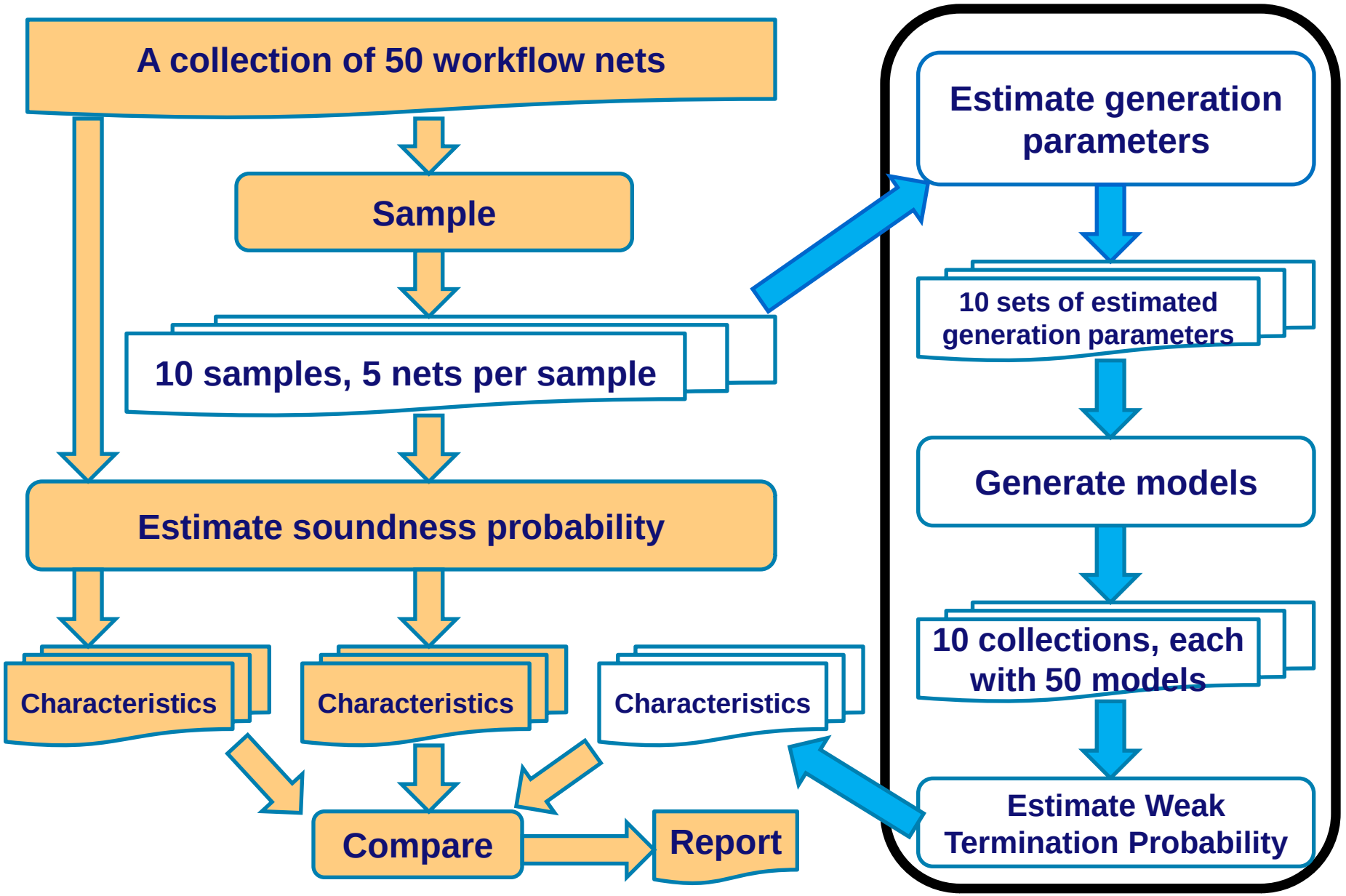


(d) Histogram of the LLP distribution in the *sample*

Result of distribution of length of the longest path (cont'd)

	80% percentile	90% percentile
original collection	102.00	107.90
generated collection 1	95.00	101.00
generated collection 2	103.80	112.00
generated collection 3	108.80	112.00
generated collection 4	104.80	111.90
generated collection 5	103.40	107.00
⋮	⋮	⋮
generated collection 20	100.80	108.00
Avg.	101.69	108.28
Std.Dev.	6.88	6.51

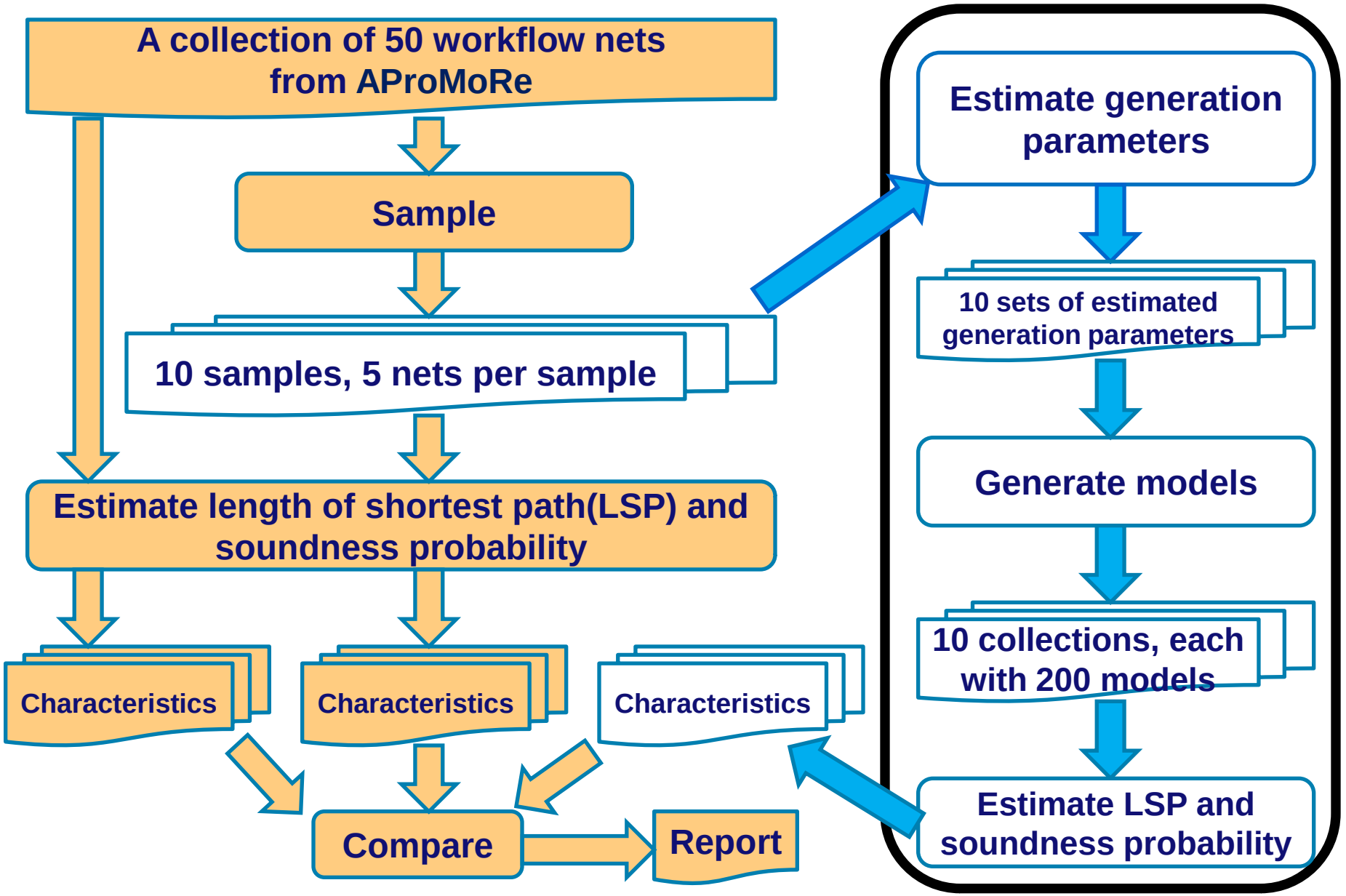
Experiments with artificial models



Probability of weak termination

	Soundness Probability
original collection	46%
sample 1	20%
generated collection 1	50%
sample 2	60%
generated collection 2	44%
sample 3	20%
generated collection 3	42%
⋮	⋮
sample 10	60%
generated collection 10	46%
Avg. of samples	46%
Std.Dev. of samples	18.97
Avg. of generated collections	45.8%
Std.Dev. of generated collections	4.57

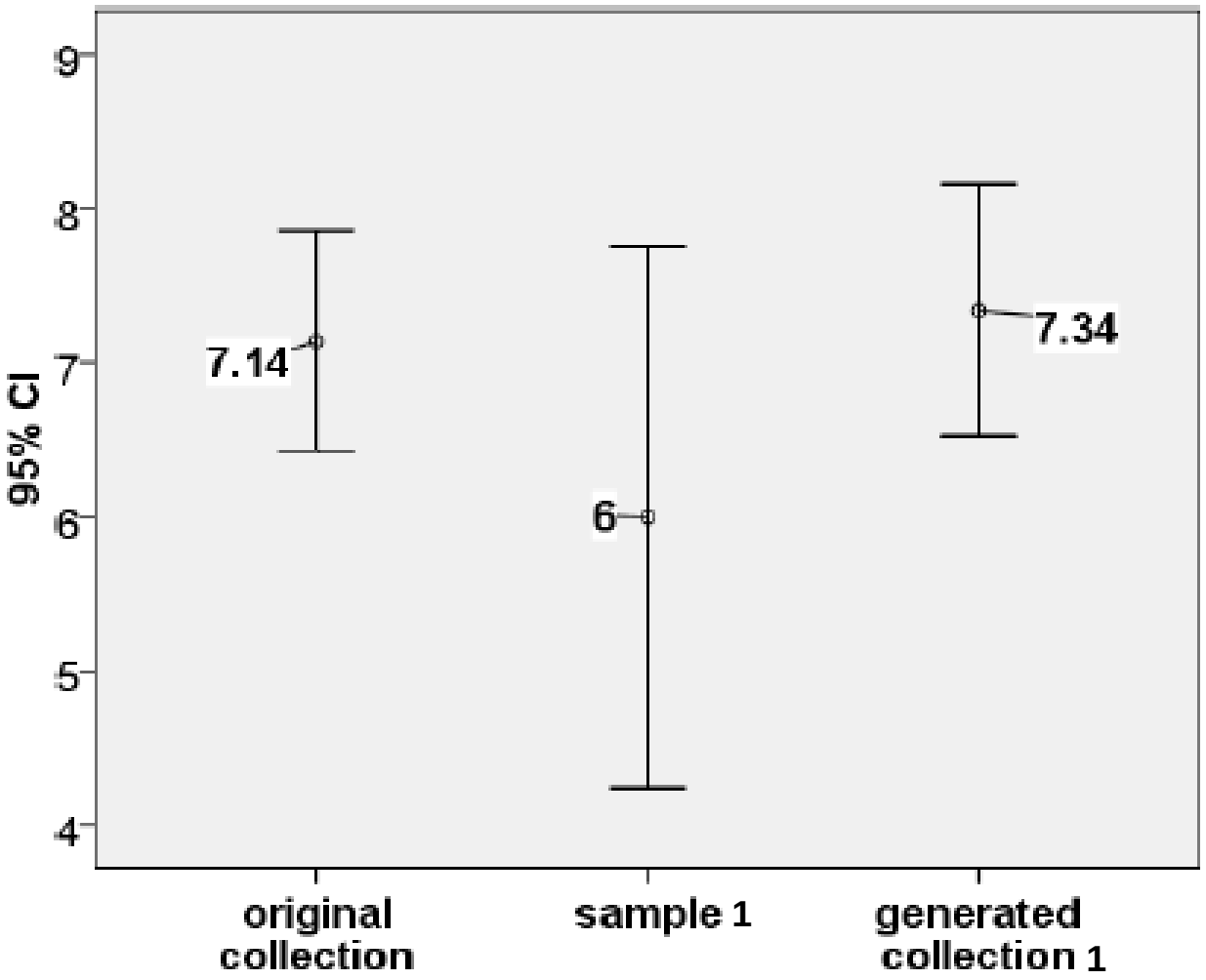
Experiments with models from industry



Result of distribution of length of the shortest path

	LSP
original collection	7.14
generated collection 1	7.34
generated collection 2	6.60
generated collection 3	6.31
⋮	⋮
generated collection 10	7.98
Avg.	7.29
Std.Dev.	1.46

Result of distribution of length of the shortest path (cont'd)



Result of soundness probability

collection	probability of soundness
original collection	1
generated collection 1	0.96
generated collection 2	1
generated collection 3	1
generated collection 4	1
generated collection 5	1
generated collection 6	0.89
generated collection 7	1
generated collection 8	1
generated collection 9	1
generated collection 10	0.85
Avg.	0.97
Std.Dev.	0.05

Conclusion

- **The assumption that workflow nets in practice are sampled from a collection seems to be realistic**
- **We are able to compute properties of a collection on basis of *small* samples while these properties cannot be computed *directly* from the samples**
- **This approach seems to be interesting to apply to collections of programs in specific application domains**

3. Towards experimental computer science

The following topics can't be studied without experiments:

- **Evidence-based best practices in software engineering and algorithmics**
- **Statistical based system testing**
- **Analysis and visualization of very large data sets**
- **Model discovery / identification from log files: process mining (special branch of datamining)**
- **Human-Machine Interaction**
- **Robots, the new platforms for CS: Interactions with physical world**

THANK YOU !

Queensland University of Technology
Brisbane Australia

Technische Universiteit
Eindhoven
University of Technology