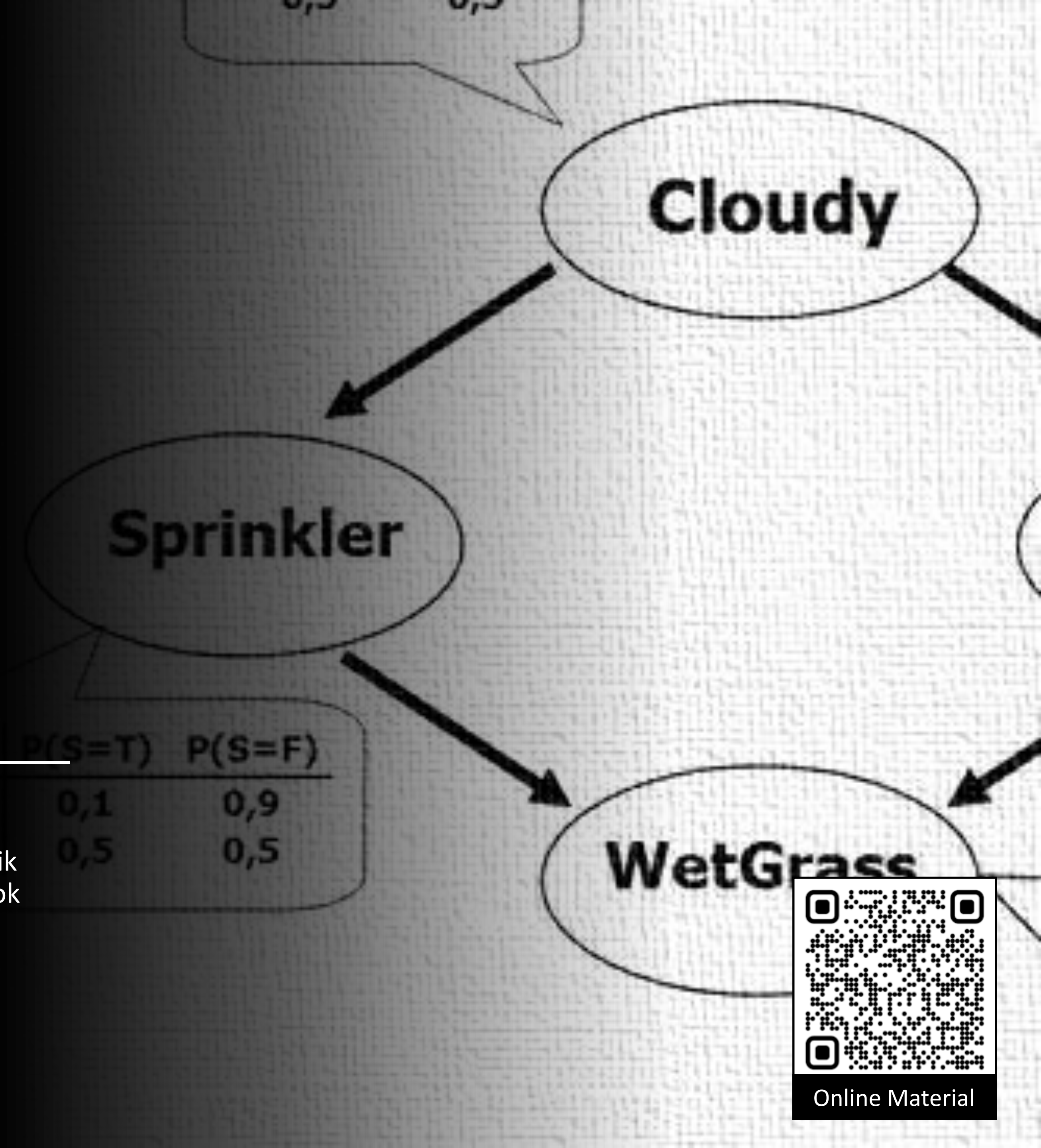
Discussion

CS 5/7320
Artificial Intelligence

Probabilistic Reasoning:
Bayesian Networks

AIMA Chapter 13

Slides by Michael Hahsler
based on slides by Svetlana Lazepnik
with figures from the AIMA textbook



Online Material

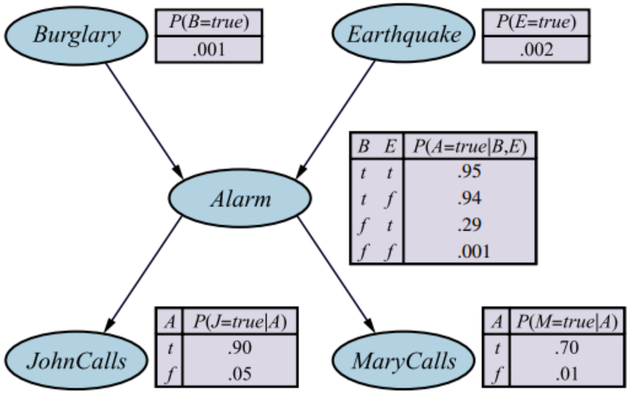
Bayesian Nets

- For a network with n boolean variables, the full joint distribution requires $O(2^n)$ probabilities.
  Example: Burglary network
  Joint probability: $2^5 - 1 = \mathbf{31}$ entries

- If each variable X_i has at most k boolean parents, then each conditional probability table (CPT) has at most 2^k rows. The CPTs for all n nodes contain then at most $O(n \times 2^k) = O(n)$ probabilities.
  Example: Burglary network
  Using CPTs: $1 + 1 + 4 + 2 + 2 = \mathbf{10}$

- **This reduces the space complexity from $O(2^n)$ to $O(n)$ and lets us store large networks!**

- **Note**: The Bayesian network stores all information needed for the complete joint probability. It let's us make optimal Bayesian decisions!



This is too much work for large networks!

Exact Inference: Calculating Conditional Distributions

Goal

- Query *variables:* X
- *Evidence* (*observed*) event: $E = e$
- *Set of unobserved* variables: Y
- Calculate the probability of X given e.

If we know the full joint distribution $P(X, E, Y)$, we can infer X by:

$$\boldsymbol{P}(X|E=e) = \frac{\boldsymbol{P}(X,e)}{P(e)} = \frac{\sum_y \boldsymbol{P}(X,e,y)}{P(e)} \propto \sum_y \boldsymbol{P}(X,e,y)$$

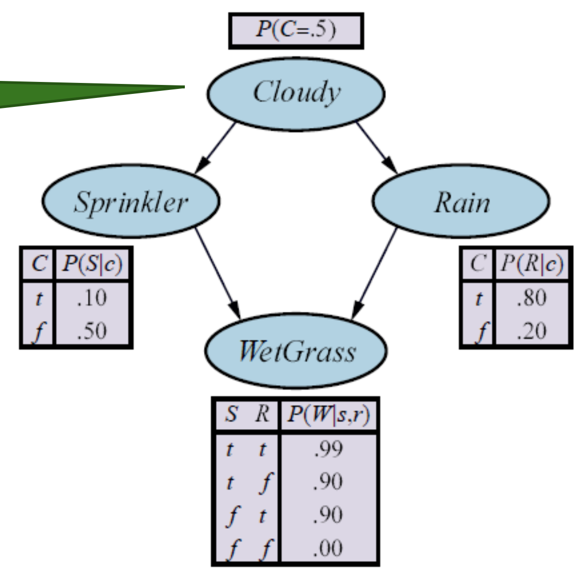
Sum over values of unobservable variables = marginalizing them out.

Approximate Inference: Prior-Sample Algorithm

function PRIOR-SAMPLE(bn) **returns** an event sampled from the prior specified by bn
  inputs: bn, a Bayesian network specifying joint distribution $\mathbf{P}(X_1, \ldots, X_n)$

  $\mathbf{x} \leftarrow$ an event with n elements
  for each variable X_i **in** $X_1, \ldots, X_n$ **do**
    $\mathbf{x}[i] \leftarrow$ a random sample from $\mathbf{P}(X_i \mid parents(X_i))$
  return x

Order is important! We need to start with the random variables that have no parents.

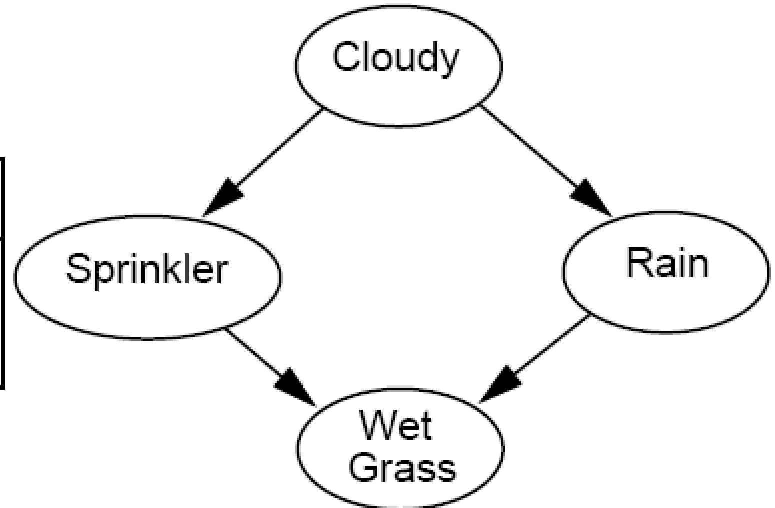


Variable Order

Note: This looks like playouts in games.

Example: Sampling from a Bayesian Network

P(C)
.50

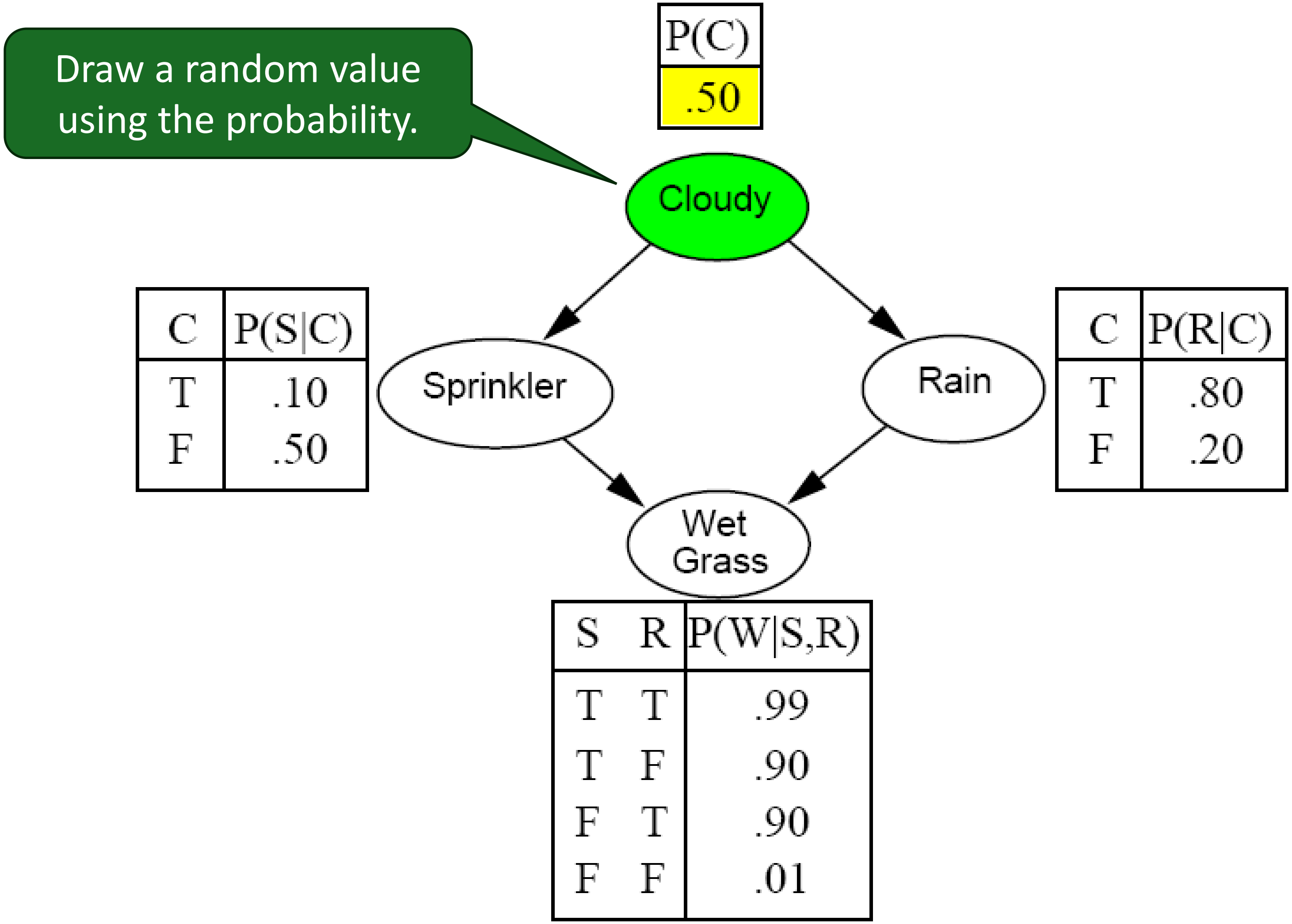


C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network



P(C)
.50

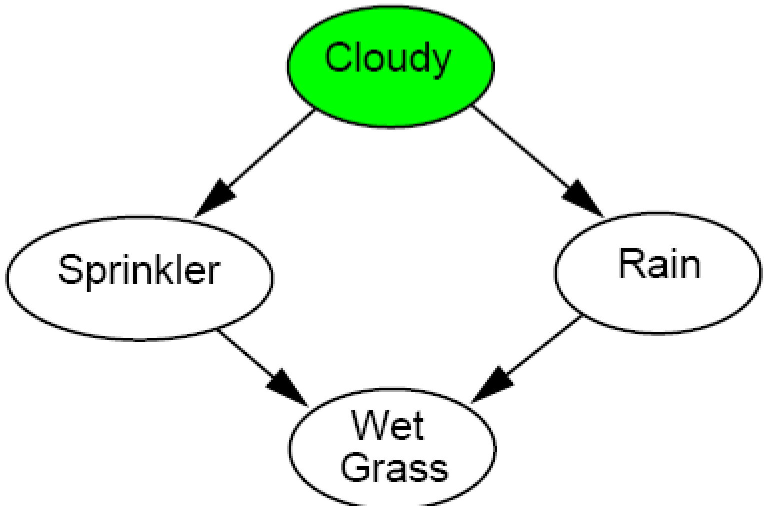
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network

P(C)
.50



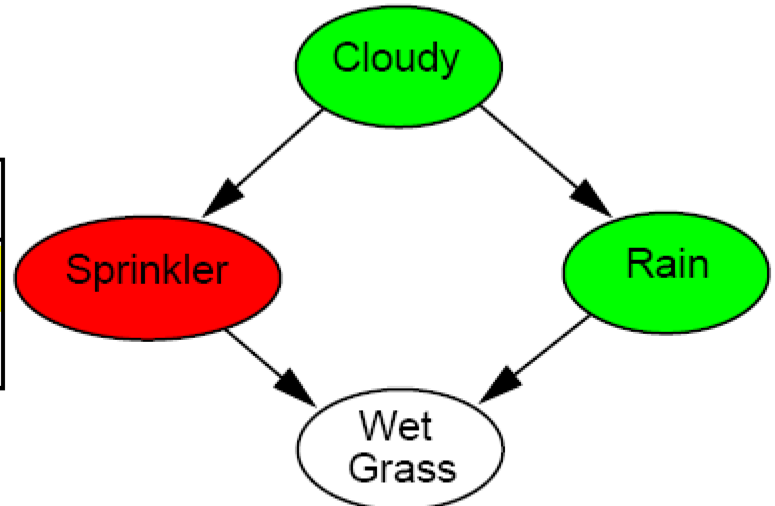
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network

P(C)
.50

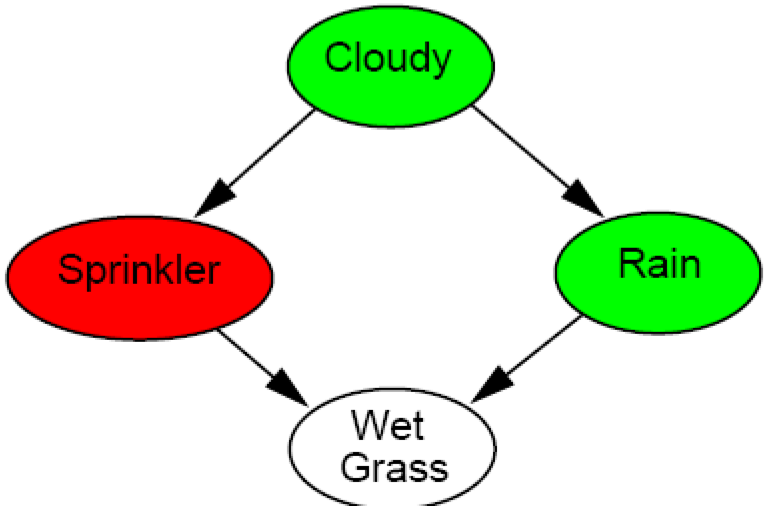


C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network



P(C)
.50

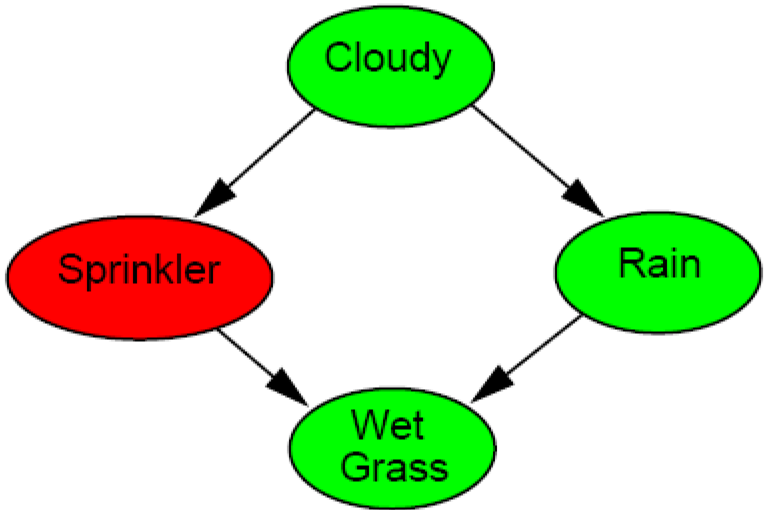
C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Example: Sampling from a Bayesian Network

P(C)
.50



C	P(S|C)
T	.10
F	.50

C	P(R|C)
T	.80
F	.20

S	R	P(W|S,R)
T	T	.99
T	F	.90
F	T	.90
F	F	.01

Prior Sample returns the event:

[C = True, S = False, R = True, W = True]

Estimating the Joint and Marginal Probability Distributions from Individual Samples

Joint Probability Distribution

Sample *N* times and determine $N_{PS}(x_1, x_2, \ldots, x_n)$, the count of how many times Prior-Sample produces event $(x_1, x_2, \ldots, x_n)$.

$$\hat{P}(x_1, x_2, \ldots, x_n) = \frac{N_{PS}(x_1, x_2, \ldots, x_n)}{N}$$

Marginal Probability Distributions

The marginal probability of a partially specified event (some x values are known) can also be calculated using the same samples. E.g.,

$$\hat{P}(x_1) = \frac{N_{PS}(x_1)}{N}$$

Estimating Conditional Probabilities: Markov Chain Monte Carlo Sampling (MCMC)

- **Idea**: Instead of creating each sample individually from scratch, **generate a sequence of samples**.
- The next sample in the sequence is created by making random changes to the current sample. Changes are controlled by a **Markov Chain** (MC) that is specifically created to have the desired probability distribution as its stationary distribution.
- The stationary distribution of a MC can be estimated using **Monte Carlo** simulation by counting how often each state (=sample) is reached in a random walk through the MC.
- Algorithms:
  1. **Gibbs sampling** works well for BNs since it needs conditional probabilities and we have CPTs.
  2. **Metropolis-Hastings** sampling is more general.

Notes:

- MCMC with Gibbs sampling is the most popular inference method.
- Simulated annealing local search is related to MCMC algorithms.

Conclusion

- Bayesian networks provide an efficient way to **store a complete probabilistic model for an AI problem** by exploiting (conditional) independence between variables.
- **Inference** means querying the model for a conditional probability given some evidence.
- Exact inference is difficult for all but tiny models.
- The state-of-the-art is to use **approximate inference** by Markov Chain Monte Carlo sampling from the model.
- Any **software libraries** provide general inference engines.