Севастопольский государственный университет
Институт информационных технологий

Методы и системы искусственного интеллекта

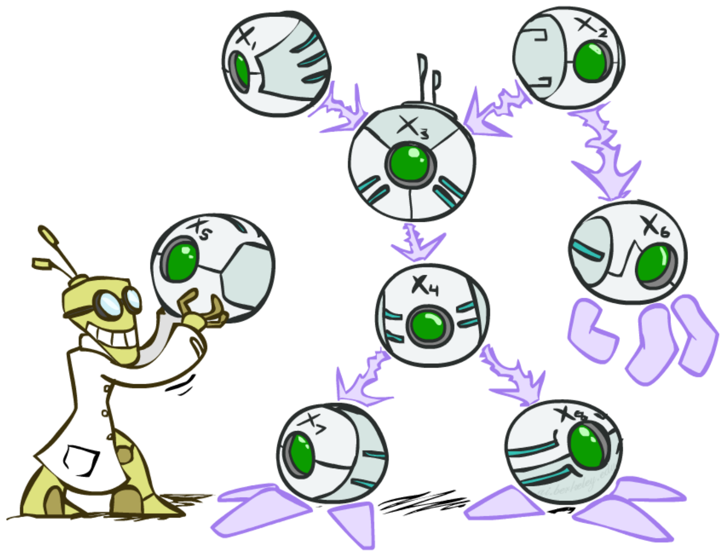
Бондарев Владимир Николаевич

Байесовские сети:
приближенный вероятностный вывод

Семантика сетей Байеса

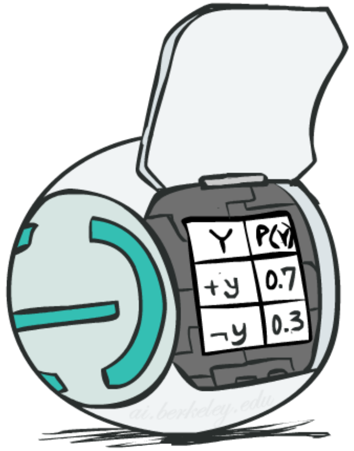
- Направленный ациклический граф, каждая вершина которого соответствует случайной переменной
- Таблица условных вероятностей (CPT) для каждой вершины:
  - Коллекция распределений X для каждой комбинации значений родительских вершин

$$P(X|a_1 \ldots a_n)$$



- Сеть Байеса неявно кодирует совместное распределение:
  - Как произведение локальных условных распределений;
  - Чтобы узнать, какая вероятность BN соответствует полному присваиванию, перемножьте все соответствующие условные вероятности :

$$P(x_1, x_2, \ldots x_n) = \prod_{i=1}^{n} P(x_i|parents(X_i))$$



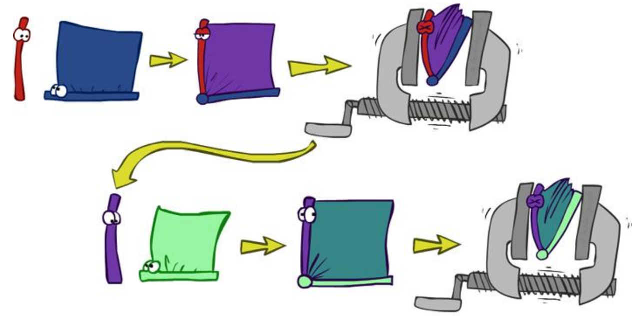
Исключение переменных

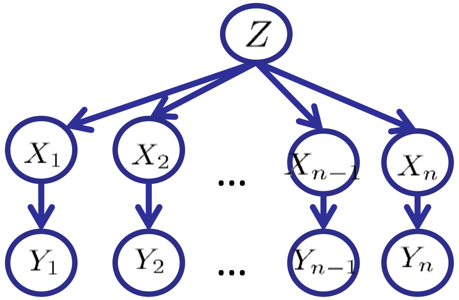
- Чередование объединения и маргинализации

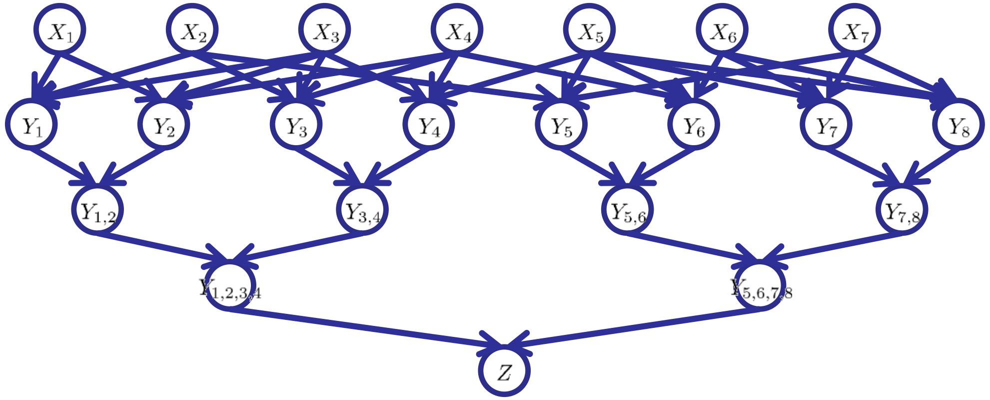
- d^k входов для фактора по k переменным с областью определения размером d

- Упорядочение переменных в ходе удаления влияет на размер формируемых факторов

Наихудший случай: экспоненциальное время выполнения по отношению к размеру сети Байеса



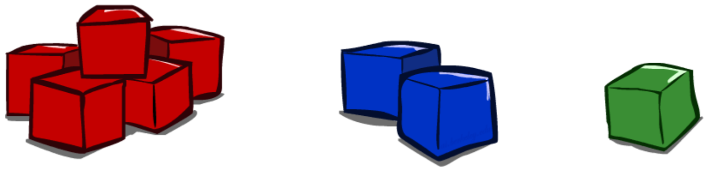




Приближенный вывод на основе выборок

Исходя из того, что точный вероятностный вывод в больших многосвязных сетях является неосуществимым, важно предусмотреть методы приближенного вероятностного вывода.

Рассмотрим рандомизированные алгоритмы выборки (называемые также алгоритмами Монте-Карло), обеспечивающие получение приближенных ответов, точность которых зависит от количества сформированных выборок.

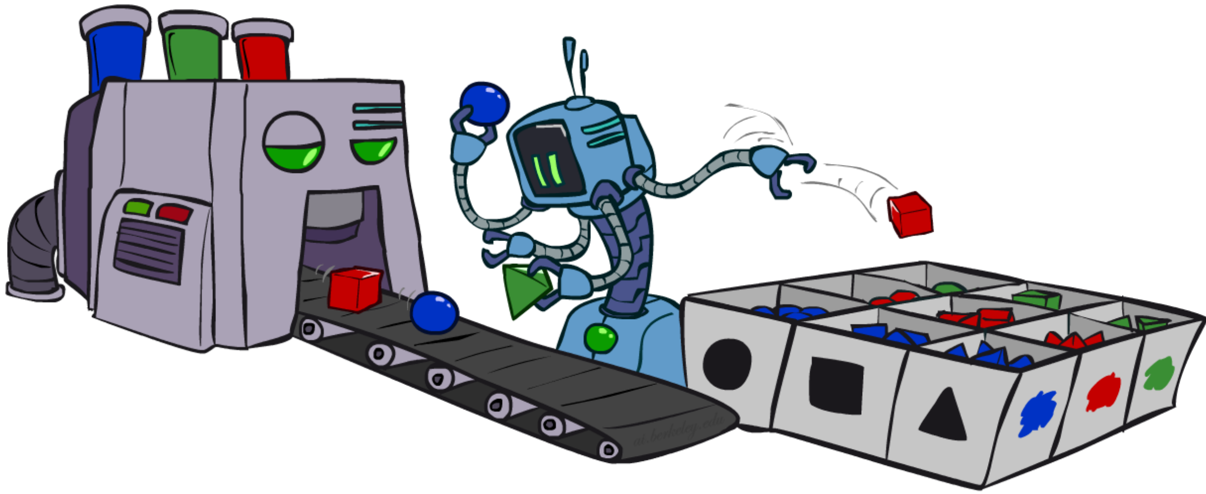


Рассмотрим применение методов выборки для вычисления апостериорных вероятностей.

Имеется два семейства алгоритмов: непосредственная выборка и выборка с помощью цепи Маркова

Непосредственные выборки

- Основная идея:
  - Взять N выборок из распределения S;
  - Вычислить приближенную апостериорную вероятность;
  - Показать, что она сходится к истинной вероятности P
- Преимущества выборок?
  - Если имеется источник случайных чисел в диапазоне (0,1) , то можно легко формировать выборки из любого желаемого распределения
  - Формирование выборок происходит быстрее, чем вычисление ответа на запрос, например, методом исключения переменных.



Непосредственные выборки

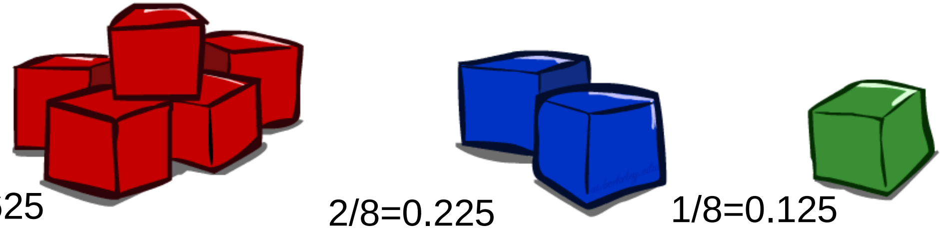
- Формирование выборки из заданного распределения
  - Шаг 1: Получить выборку *u* из равномерного распределения в интервале [0, 1)
    - Например, random() в Пайтон
  - Шаг 2: Преобразовать эту выборку *u* в результирующую, связав каждое формируемое значение случайной переменной с размером поддиапазона, равным требуемой вероятности появления соответствующего значения.

- Пример

C	P(C)
red	0.6
green	0.1
blue	0.3

$$0 \le u < 0.6, \to C = red$$
$$0.6 \le u < 0.7, \to C = green$$
$$0.7 \le u < 1, \to C = blue$$

- Если random() возвращает *u* = 0.83, то выборное значение *C* = blue
- Например, после 8 выборок можно получить набор выборочных значений со следующими вероятностями:

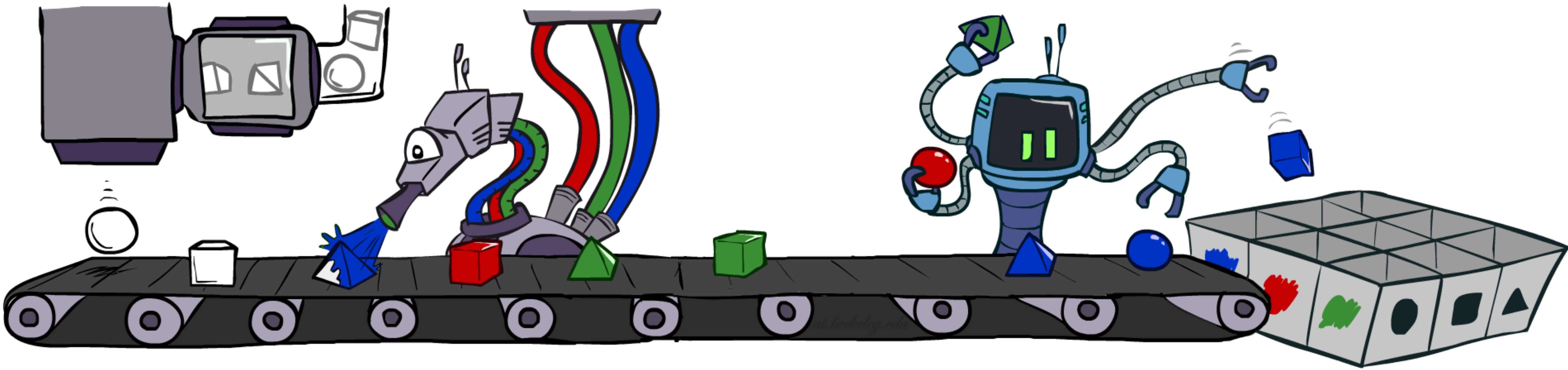


5/8=0.625 2/8=0.225 1/8=0.125

Выборочные методы в сетях Байеса

1. Выборки из априорных распределений
2. Формирование выборок путем исключения (Rejection Sampling)
3. Формирование выборок с учетом правдоподобия

Выборки из априорных распределений

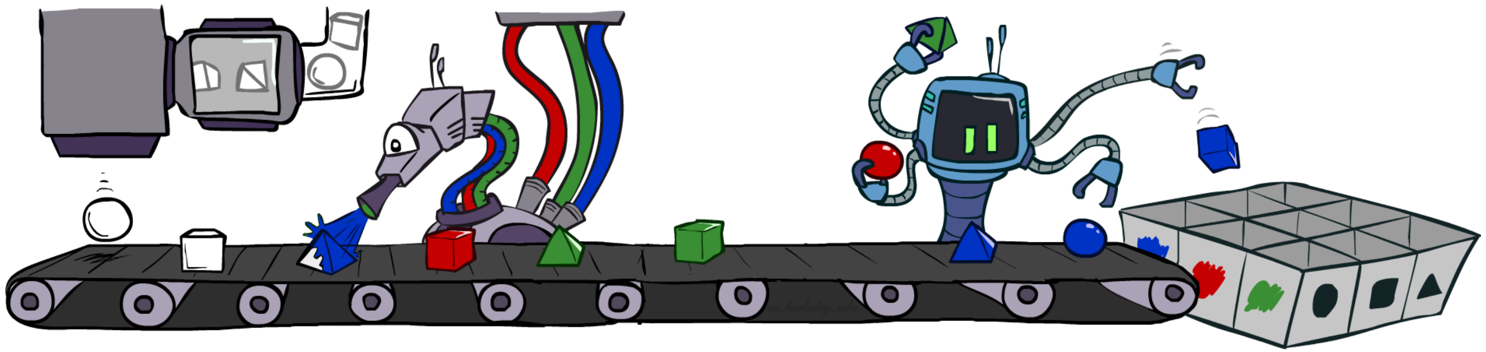


Алгоритм Prior-Sample (PS) для формирования выборки

function Prior-Sample(bn) **returns** событие, выработанное путем применения операции формирования выборки к априорному распределению, заданному в виде сети bn
    inputs: bn, байесовская сеть, задающая совместное распределение $\mathbf{P}(X_1, \ldots, X_n)$

    $x \leftarrow$ событие с n элементами
    for i = 1 **to** n **do**
        $x_i \leftarrow$ случайная выборка из $\mathbf{P}(X_i | parents(X_i))$
    return x

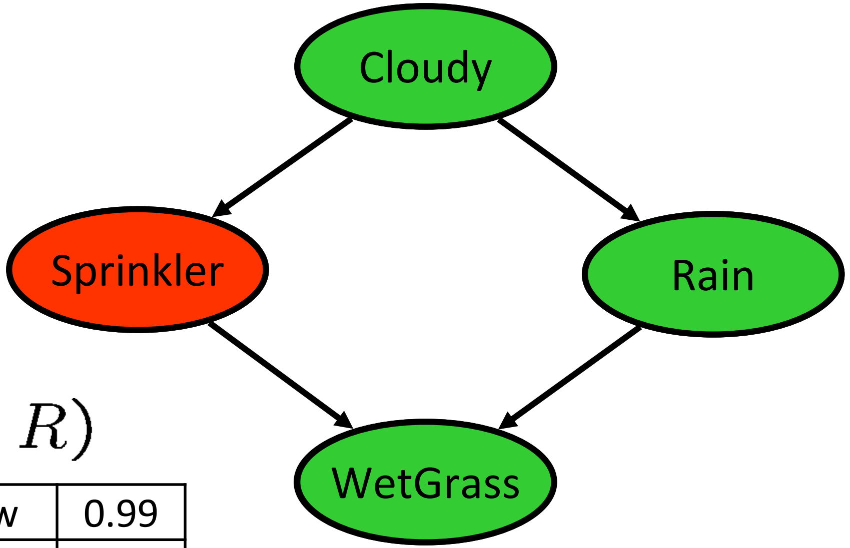
1. Формируются события, не имеющие связанных с ними свидетельств;
2. Выборка формируется последовательно по каждой переменной, **в топологическом порядке.** Распределение, из которого берется выборка значения, обусловливается значениями, уже присвоенными родительским переменным.



Выборки из априорных распределений

$P(C)$

+c	0.5
-c	0.5



$P(S|C)$

+c	+s	0.1
	-s	0.9
-c	+s	0.5
	-s	0.5

$P(R|C)$

+c	+r	0.8
	-r	0.2
-c	+r	0.2
	-r	0.8

$P(W|S,R)$

+s	+r	+w	0.99
		-w	0.01
	-r	+w	0.90
		-w	0.10
-s	+r	+w	0.90
		-w	0.10
	-r	+w	0.01
		-w	0.99

Имеет место упорядочение
[Cloudy, Sprinkler, Rain, WetGrass]

Выборки из распределений:

+c, -s, +r, +w

-c, +s, -r, +w

…

Prior-Sample возвращает событие
[true, false, true, true] с вероятностью:

S_{ps}({**+c, -s, +r, +w** }) =
0.5x0.9x0.8x0.9 = 0.324

Выборки из априорных распределений

- Этот процесс генерирует выборки с вероятностью:

$$S_{PS}(x_1 \ldots x_n) = \prod_{i=1}^{n} P(x_i|\text{Parents}(X_i)) = P(x_1 \ldots x_n)$$

…т.е. с совместной BN вероятностью.

- Пусть число выборок для события будет $N_{PS}(x_1 \ldots x_n)$
  Тогда

$$\begin{aligned}\lim_{N\to\infty} \hat{P}(x_1, \ldots, x_n) &= \lim_{N\to\infty} N_{PS}(x_1, \ldots, x_n)/N \\ &= S_{PS}(x_1, \ldots, x_n) \\ &= P(x_1 \ldots x_n)\end{aligned}$$

- т.е., оцениваемая вероятность становится точной при большом числе выборок.
  Такая оценка называется **согласованной.**

Пример

- Получим набор выборок для BN:

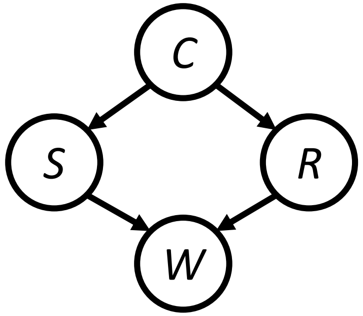
  +c, -s, +r, +w

  +c, +s, +r, +w

  -c, +s, +r, -w

  +c, -s, +r, +w

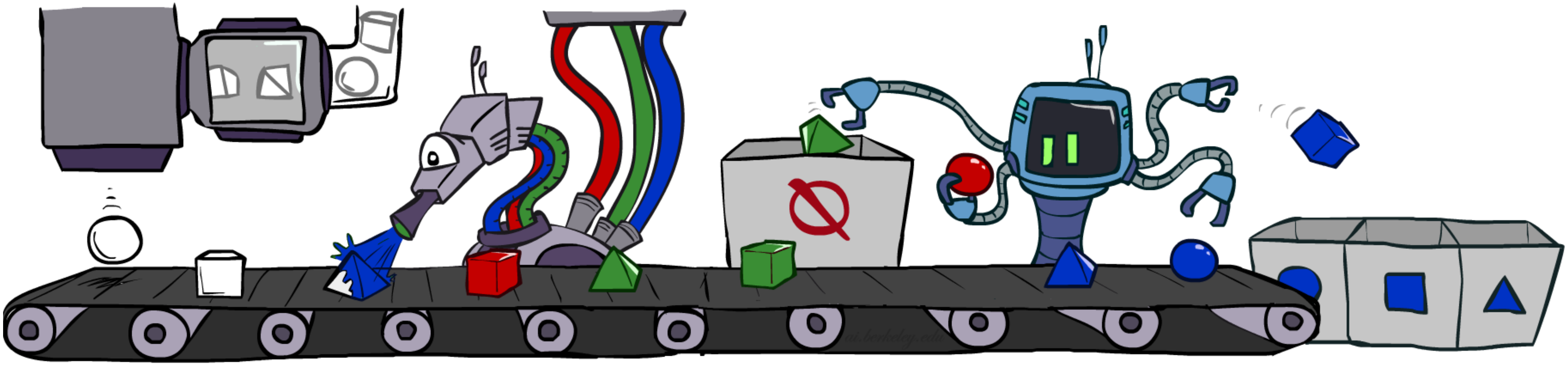
  -c, -s, -r, +w



- Если мы хотим знать P(W):
  - Мы должны подсчитать число событий <+w:4, -w:1>;
  - Нормализовать, чтобы получить P(W) = <+w:0.8, -w:0.2>;
  - Эти значения будут тем ближе к истинному распределению, чем больше выборок;

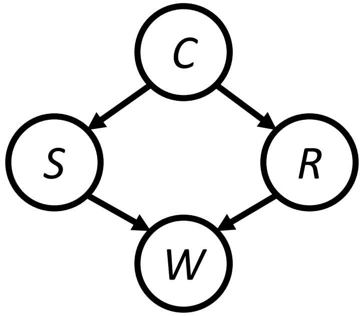
  - Можем также оценить и другие вероятности
    - P(C | +w)?   P(C | +r, +w)?

Формирование выборок с исключением в BN



Формирование выборок с исключением (Rejection-Sampling)

- Пусть требуется вычислить P(C):
  - Просто подсчитайте число событий C во время генерации значений.
- Пусть требуется вычислить P(C | +s):
  - То же самое: подсчитайте число C, но игнорируйте (отклоняйте) выборки, для которых не выполняется S = + s;
  - Это называется **исключением выборок,** т.е. **исключаются все выборки, которые не соответствуют свидетельству;**
  - Позволяет исключать выборки заранее;
  - Подход обеспечивает согласованную оценку условных вероятностей (т.е. верно в пределе).

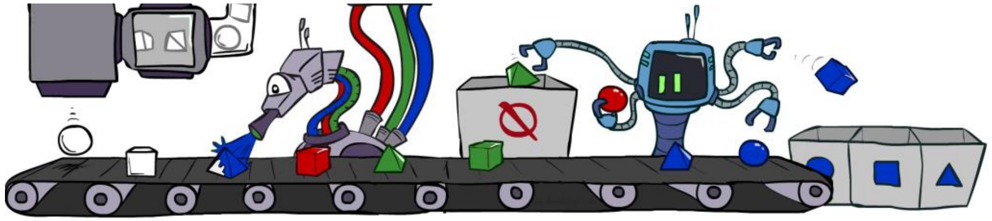


+c, -s, +r, +w
+c, +s, +r, +w
-c, +s, +r, -w
+c, -s, +r, +w
-c, -s, -r, +w

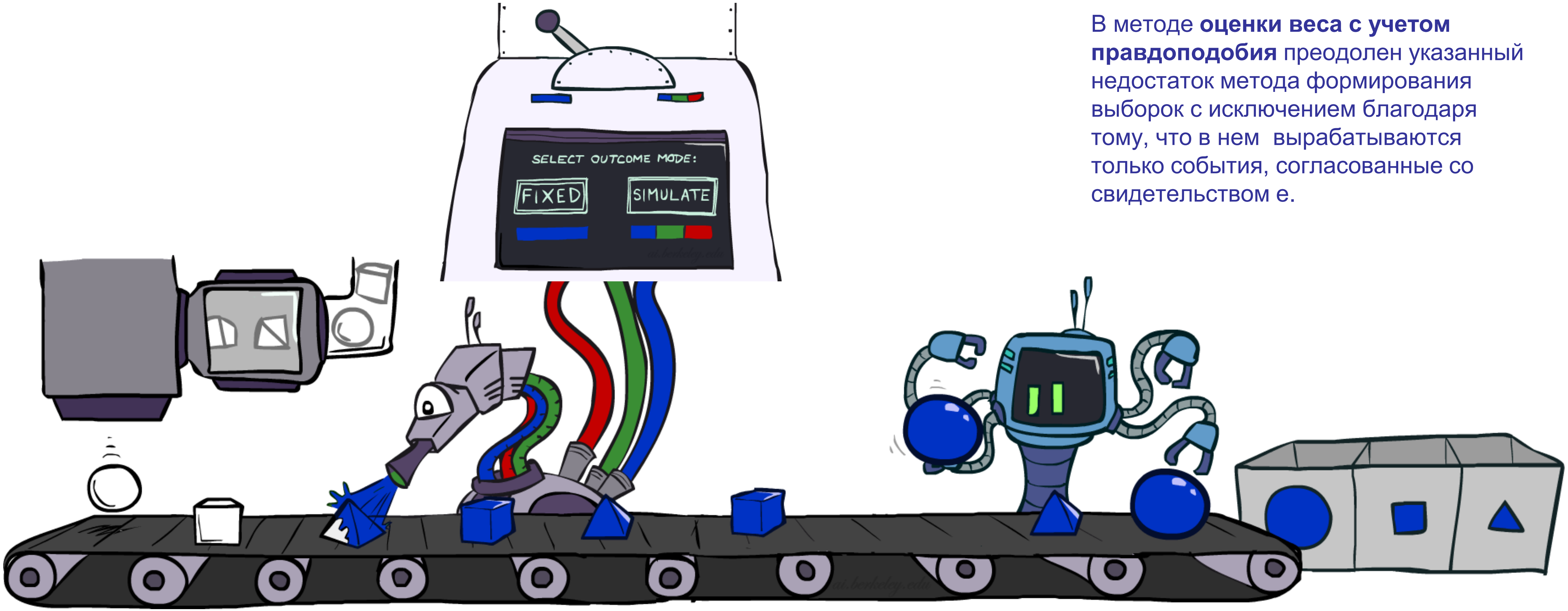
Алгоритм Rejection-Sampling

```
function Rejection-Sampling(X, e, bn, N) returns оценка значения P(X|e)
    inputs: X, переменная запроса
            e, свидетельство, определяемое как некоторое событие
            bn, байесовская сеть
            N, общее количество выборок, которые должны
               быть сформированы
    local variables: N, вектор результатов подсчетов по X,
                        первоначально равный нулю

    for j = 1 to N do
        x ← Prior-Sample(bn)
        if выборка x согласуется со свидетельством e then
            N[x] ← N[x]+1, где x представляет собой значение
                           переменной X в множестве x
    return Normalize(N[X])
```

Самым большим недостатком алгоритма формирования выборок с исключением является то, что в нем приходится исключать слишком много выборок!



Оценка веса с учетом правдоподобия



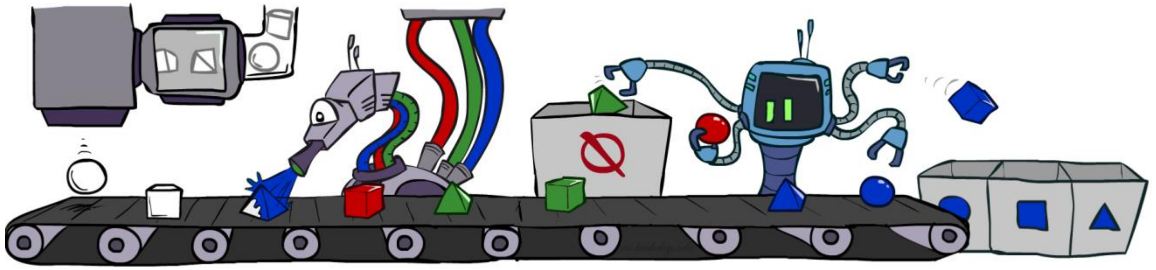
В методе **оценки веса с учетом правдоподобия** преодолен указанный недостаток метода формирования выборок с исключением благодаря тому, что в нем вырабатываются только события, согласованные со свидетельством e.

Оценка веса с учетом правдоподобия

- Проблемы с исключением выборок:
  - Если свидетельство маловероятно, то отклоняется большое число выборок
  - Рассмотрим P(Shape | blue)

Shape → Color

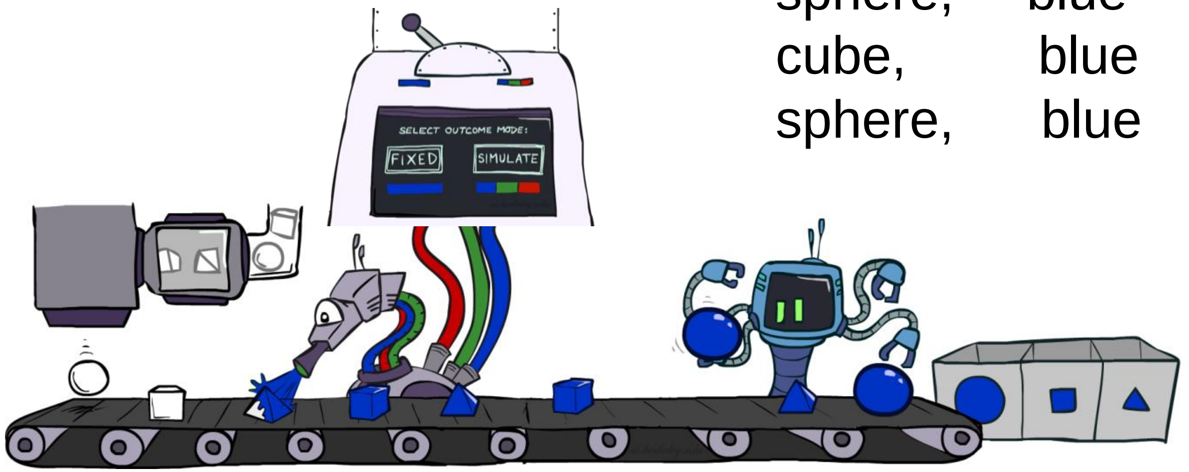
~~pyramid, green~~
~~pyramid, red~~
sphere, blue
~~cube, red~~
~~sphere, green~~



- Идея: зафиксировать значения свидетельств и выполнить выборки оставшихся переменных
  - Проблема: выборочное распределение не согласовано!
  - Решение: взвесить каждую выборку с учетом вероятности свидетельств

Shape → Color

pyramid, blue
pyramid, blue
sphere, blue
cube, blue
sphere, blue



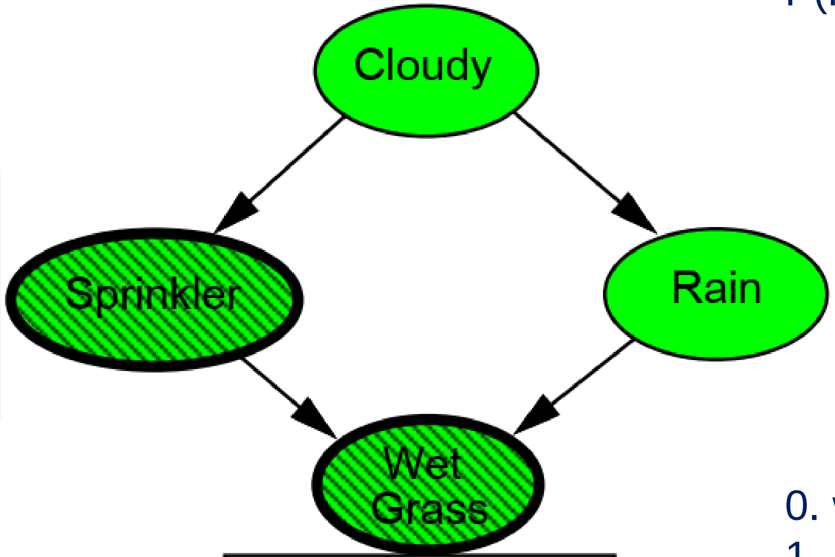
Оценка веса с учетом правдоподобия

```
function Likelihood-Weighting(X, e, bn, N) returns оценка
        значения P(X|e)
    inputs: X, переменная запроса
            e, свидетельство, определяемое как некоторое событие
            bn, байесовская сеть
            N, общее количество выборок, которые должны
               быть сформированы
    local variables: W, вектор взвешенных результатов подсчетов по X,
                        первоначально равный нулю

    for j = 1 to N do
          x, w ← Weighted-Sample(bn)
          W[x] ← W[x]+w, где x представляет собой значение переменной X
                            в множестве x
    return Normalize(W[X])

function Weighted-Sample(bn, e) returns событие x и вес w

    x ← событие с n элементами; w ← 1
    for i = 1 to n do
          if Xi имеет значение xi in свидетельство e
                then w ← w×P(Xi=xi|parents(Xi))
                else xi ← случайная выборка из P(Xi|parents(Xi))
    return x, w
```

Каждое событие взвешивается с учетом правдоподобия того, что оно согласуется свидетельством. Такое правдоподобие измеряется с помощью произведения условных вероятностей для каждой переменной свидетельства, если даны ее родительские переменные. Интуитивно ясно, что события, в которых фактическое свидетельство кажется маловероятным, должны получать меньший вес.

Пример

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

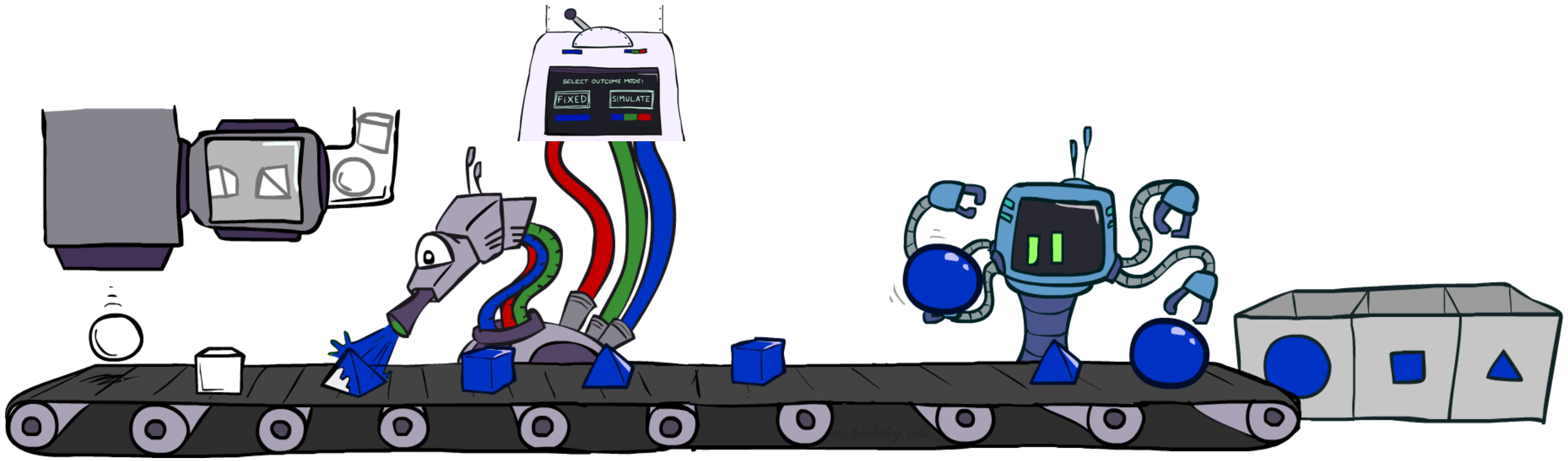
Запрос:
P(Rain | Sprinkler=true, WetGrass=true)

0. w=1:
1. Пусть выборка C=true;
2. Дано S=true, w ← w * P(S=true | C=true) = w*0.1;
3. Пусть выборка R=true;
4. Дано W=true, w ← w*P(W=true | S=true, R=true)=w*0.99=0.099

Алгоритм возвращает событие [true,true,true,true] с весом 0.099

Оценка веса с учетом правдоподобия

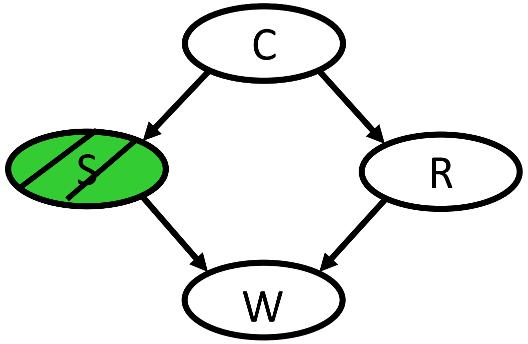
- Вход: значения свидетельств
- w = 1.0
- for i = 1, 2, …, n в топологическом порядке
  - if X_i переменная свидетельства
    - X_i = наблюдение x_i для X_i
    - w = w * P(x_i | Parents(X_i))
  - else
    - Выборка x_i из P(X_i | Parents(X_i))
- return (x_1, x_2, …, x_n), w



Оценка веса с учетом правдоподобия

- Напомним, что переменные свидетельств **E** зафиксированы со значениями e_i. Обозначим все прочие переменные как **Z**. Выборки для каждой переменной из **Z**, если даны ее родительские значения, формируются следующим образом:

$$S_{WS}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{l} P(z_i|\text{Parents}(z_i))$$



- При этом на значения сформированных выборок для каждой переменной Z_i кроме других предков Z_i могут оказать влияние предки-свидетельства.

- Веса выборок с учетом вероятностей свидетельств:

$$\mathrm{w}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{m} P(e_i|\text{Parents}(e_i))$$

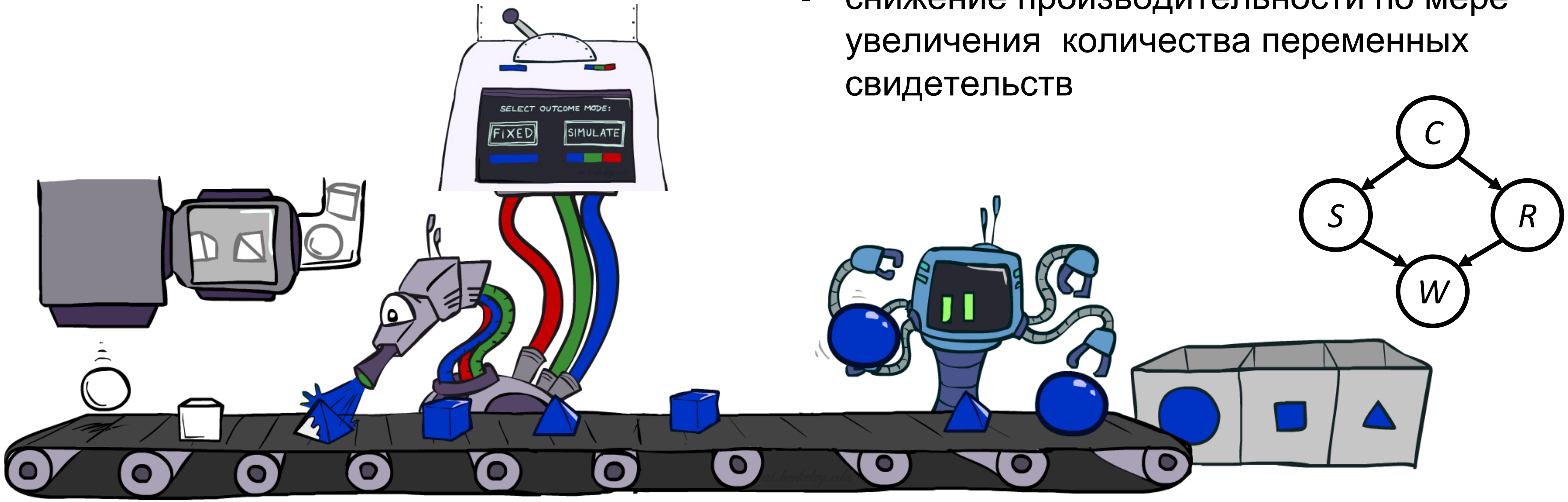
- Объединяя, получаем согласованное взвешенное выборочное распределение:

$$P(z_1 \ldots z_p, e_1 \ldots e_m) = S_{WS}(\mathbf{z}, \mathbf{e})\, \mathrm{w}(\mathbf{z}, \mathbf{e}) = \prod_{i=1}^{l} P(z_i|\text{Parents}(z_i)) \prod_{i=1}^{m} P(e_i|\text{Parents}(e_i)) = P(\mathrm{z}, \mathrm{e})$$

Оценка веса с учетом правдоподобия

- Взвешивание с учетом правдоподобия полезно, т.к.:
  - Учитывает свидетельства при генерации выборок;
  - Например. значение W выбиралось на основе значений свидетельств S, C;
  - Большинство выборок отражает состояние мира, определяемого свидетельствами.

- Взвешивание с учетом правдоподобия не решает всех проблем:
  - Свидетельства влияют на выбор нижестоящих переменных, но не на вышестоящие (вероятность того, что C получит значение, совпадающее со свидетельствами, невысока);
  - снижение производительности по мере увеличения количества переменных свидетельств



Алгоритм Монте-Карло с применением цепи Маркова (MCMC – Markov chain Monte Carlo)

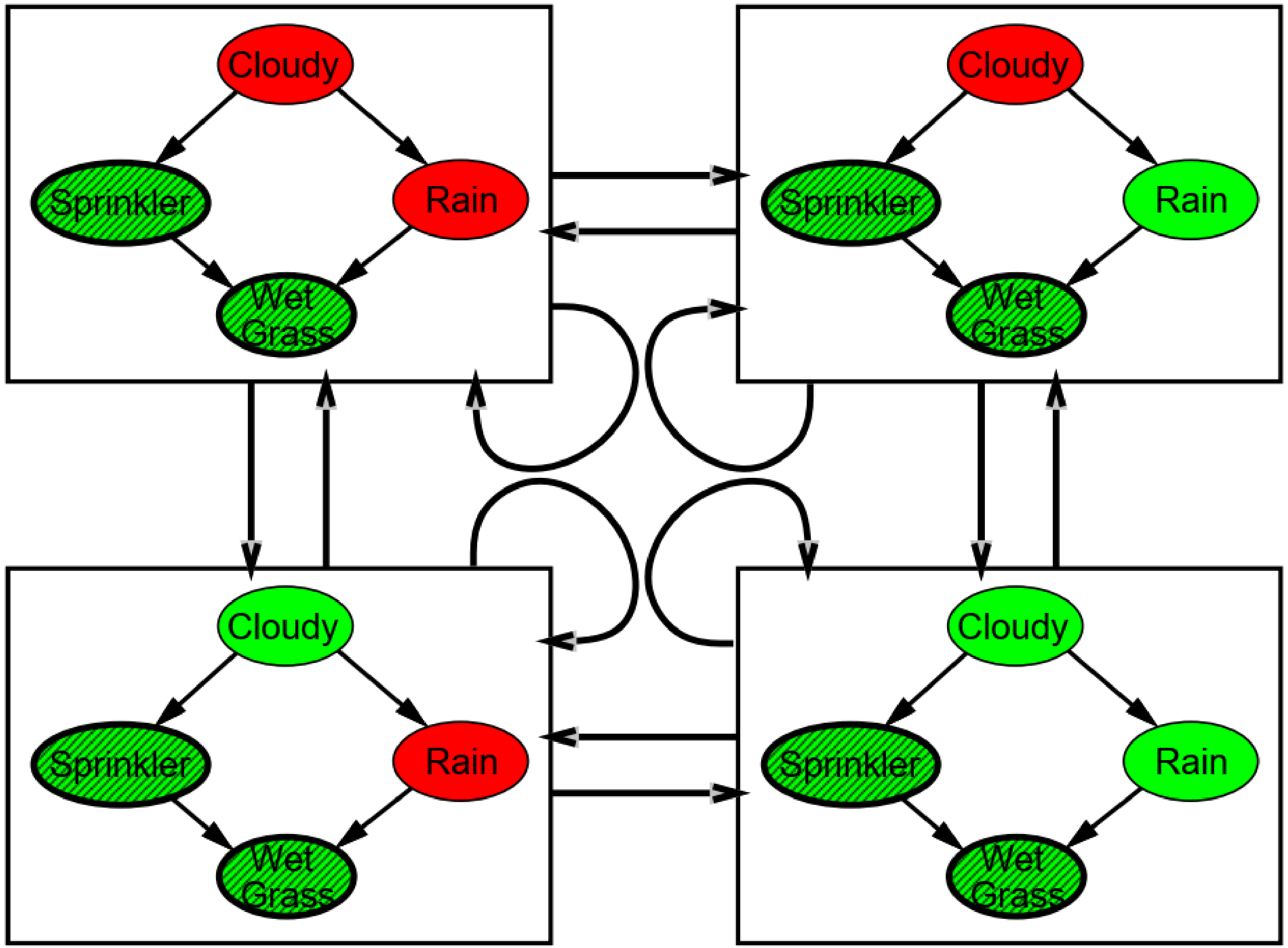
- В отличие от рассмотренных алгоритмов формирования выборок, которые вырабатывают каждое событие с нуля, в алгоритме MCMC **каждое событие вырабатывается путем внесения случайного изменения в предыдущее событие**.
- Поэтому целесообразно рассматривать сеть, находящуюся в конкретном текущем **состоянии**, заданном с помощью присваивания значений каждой переменной.
- Переход в следующее состояние осуществляется путем случайного формирования выборки значения для одной из переменных X_i, отличных от переменных свидетельства, причем это значение обусловлено текущими значениями переменных в Марковском покрытии переменной X_i.
- **Марковского покрытие (Markov blanket) переменной** состоит из ее родительских переменных, дочерних переменных и родительских переменных дочерних переменных.
- В алгоритме MCMC предусмотрено случайное блуждание по пространству состояний (пространству возможных полных присваиваний), в котором каждый раз изменяется значение одной переменной, но значения переменных свидетельств остаются фиксированными.

Алгоритм Монте-Карло с применением цепи Маркова (MCMC)

Рассмотрим запрос P(Rain|*Sprinkler=true, WetGrass=true).* 4 состояния сети. Каждое состояние - выборка:

Выборка для Cloudy с учетом текущих значений переменных Марковского покрытия
P (Cloudy| Sprinkler=true,
Rain= false)

2. [false,true,false,true]



Выборка из Rain…
P (Rain| Cloudy=false,
Sprinkler=true,
WetGrass=true)

3. [false,true,true,true]

Скрытые переменные инициализированы случайным образом

1. [true,true,false,true]
C S R W

Выборка из Cloudy…
P (Cloudy| Sprinkler=true,
Rain= false)

4. [true,true,true,true]

Алгоритм Монте-Карло с применением цепи Маркова (MCMC)

```
function MCMC-Ask(X, e, bn, N) returns оценка значения P(X|e)
    local variables: N[X], вектор результатов подсчетов по X,
                           первоначально равный нулю
                     Z, переменные в сети bn, отличные от переменных
                        свидетельства
                     x, текущее состояние сети, первоначально
                        скопированное из e

    инициализировать x случайными значениями переменных из Z
    for j = 1 to N do
        for each Zi in Z do
          выполнить выборку значения переменной Zi в векторе x
             из P(Zi|mb(Zi)), если даны значения MB(Zi) в векторе x
          N[x] ← N[x]+1, где x представляет собой значение переменной X
                           в множестве x
     return Normalize(N[X])
```

MCMC, Пример

Оценим P(Rain|*Sprinkler=true, WetGrass=true).*

Повторяем случайные выборки значений для переменных *Rain и Cloudy* с учетом их марковских покрытий.
Подсчитываем число случаев для Rain=true и Rain=false .

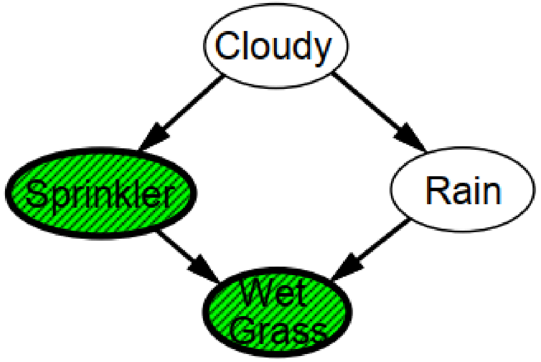
Если в данном процессе посещаются 20 состояний, в которых переменная Rain=true, и 60 состояний, в которых переменная Rain=false, то

P(Rain|*Sprinkler=true, WetGrass=true)*=Normalize(<20, 60>) =<0 .25,0. 75>

Теорема: *процесс формирования выборок переходит в "динамическое равновесие", в котором в конечном итоге доля времени, проведенного в каждом состоянии, точно пропорциональна апостериорной вероятности этого состояния.*

Выборки с учетом Марковского покрытия

Марковское покрытие для *Cloudy – Sprinkler* и *Rain*
Марковское покрытие для *Rain – Cloudy, Sprinkler* и *WetGrass*



Вероятность с учетом Марковского покрытия

$$P(x_i'|mb(X_i)) = P(x_i'|parents(X_i))\prod_{Z_j \in Children(X_i)} P(z_j|parents(Z_j))$$

Основные вычислительные проблемы алгоритма:
1) Трудно оценить, достигнута ли сходимость;
2) Может быть затратным, если Марковское покрытие большое:

$P(X_i|mb(X_i))$ не будет сильно изменяться (закон больших чисел).

Заключение

Точный вывод методом исключения переменных:
- полиномиальное время на полидеревьях, NP-сложный на общих графах;
- пространственная сложность ~ временная сложность, очень чувствителен к топологии;

Вывод с аппроксимацией (LW, MCMC):
- LW плохо работает, когда есть много (нижестоящих) свидетельств;
- LW, MCMC вообще нечувствителен к топологии;
- сходимость может быть очень медленной к вероятностям, близким к 1 или 0;
- может обрабатывать произвольные комбинации дискретных и непрерывных переменных.