Prism Theory

Vortrag

Jonida Saqe

13. Juli 2007

Übersicht

- Markovketten (DTMC)
- PCTL
- PCTL Modellprüfung
- Fallstudie IPv4

Time-Discrete Markov Chains(DTMC)

Time-Discrete Markov Chains(DTMC)

- Markovkette in diskreter Zeit(ab jetzt mit DTMC abgekürzt): Familie zufälliger Variablen $\{X(k) \mid k = 0, 1, 2..\}$
- $X(k)$: Beobachtungen in bestimmten Zeitschritten
- **Zustände** : die von $X(k)$ entnommenen Werte
- **Zustandsraum** : Menge aller möglichen Zustände

Wir werden uns ausschließlich mit endlichen Zustandsräumen beschäftigen.

Die Markoveigenschaft

Eine DTMC erfüllt die **Markoveigenschaft**(gedächnislose Eigenschaft), wenn der Zustand der DTMC im Zeitschritt k nur von deren Zustand im Zeitschritt $k-1$ abhängt.

Für jede ganze Zahl $k \geq 0$ und die Zustände $s_0, ..., s_k$ gilt:

$P[\ X(k) \mid X(k-1) = s_{k-1}, ..., X(0) = s_0\ ] = P[\ X(k) = s_k \mid X(k-1) = s_{k-1}\ ]$

DTMC

Eine **DTMC** ist ein Tupel $\mathcal{D} = (S, \bar{s}, P)$, wobei:

- S eine endliche Zustandsmenge
- $\bar{s} \in S$ der Anfangszustand und
- $P : S \times S \to [0, 1]$ Übergangswahrscheinlichkeismatrix ist, wobei :

$$\sum_{s' \in S} P(s, s') = 1 \text{ für alle Zustände } s \in S.$$

- $\Rightarrow$ Terminierende Zustände: Eigenschleife(Übergang zum selben Zustand mit der Wahrscheinlichkeit 1).

DTMC

- Kennzeichnungsfunktion für Eigenschaftsangaben von Zuständen

$$\mathcal{L} : S \to 2^{AP},$$

die Zustände zu Mengen atomarer Aussagen einer Menge AP zusammenfasst.

- Kostenfunktion

$$C : S \times S \to \mathbb{R}^{\geq 0},$$

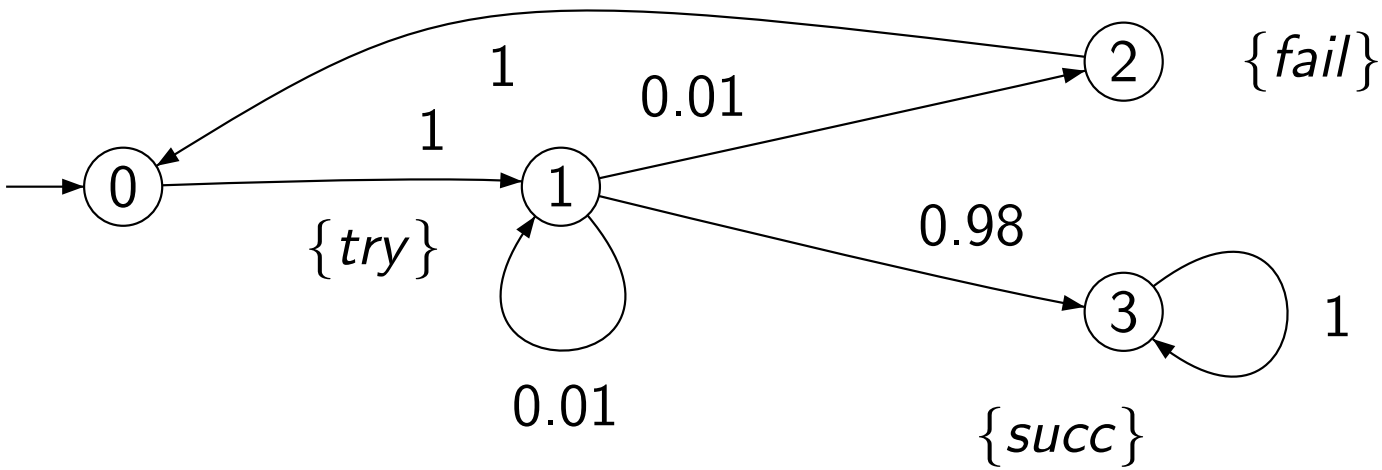
  - $C(s, s')$ hingenommenen Kosten, um vom Zustand s zu s' zu kommen
  - Beispiel: stellt Energieverbrauch, Anzahl gesendeter Packete oder Anzahl verlorener Kunden dar.

DTMC

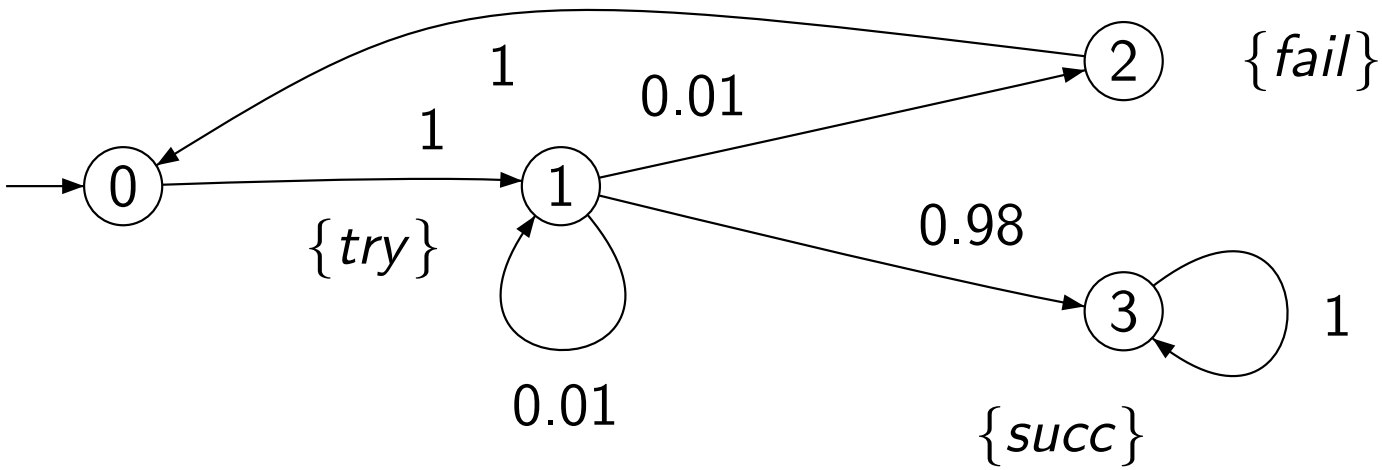
Eine mit **Aufschriften versehene Markovkette** ist also ein Tupel $(\mathcal{D}, \mathcal{L}, \mathbf{C})$, wobei:

- $\mathcal{D} = (S, \bar{s}, \mathcal{P})$ eine DTMC,
- $\mathcal{L} : S \to 2^{AP}$ die Kennzeichnungsfunktion,
- $C : S \times S \to \mathbb{R}^{\geq 0}$ die Kostenfunktion ist.

Beispiel einer beschrifteten DTMC:



Beispiel 1

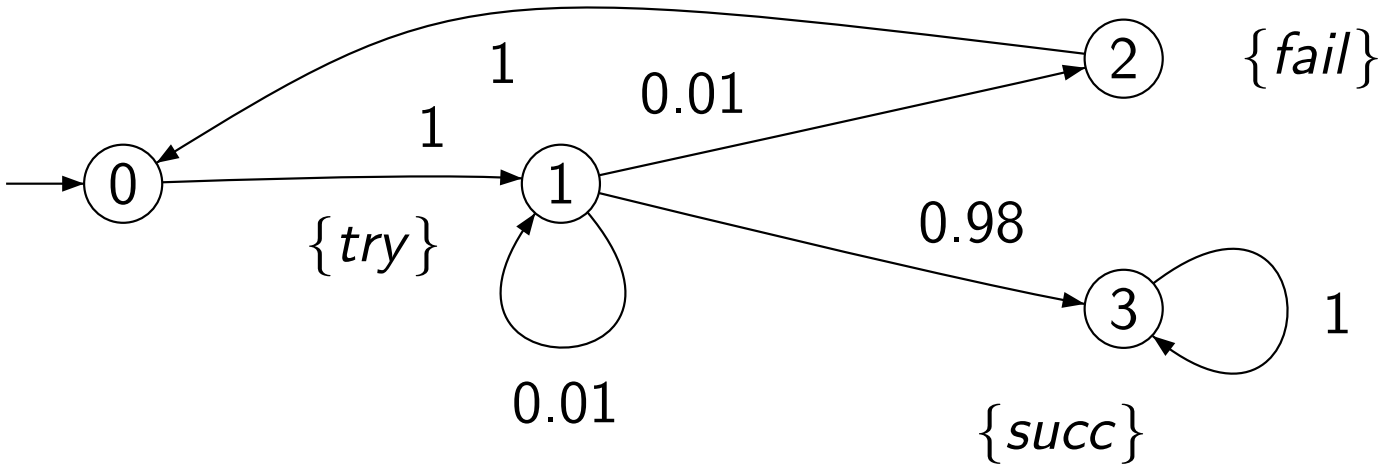


- Matrix **P** der Übergangswahrscheinlichkeiten :

$$P = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0.01 & 0.01 & 0.98 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Versuch eine Nachricht zu senden

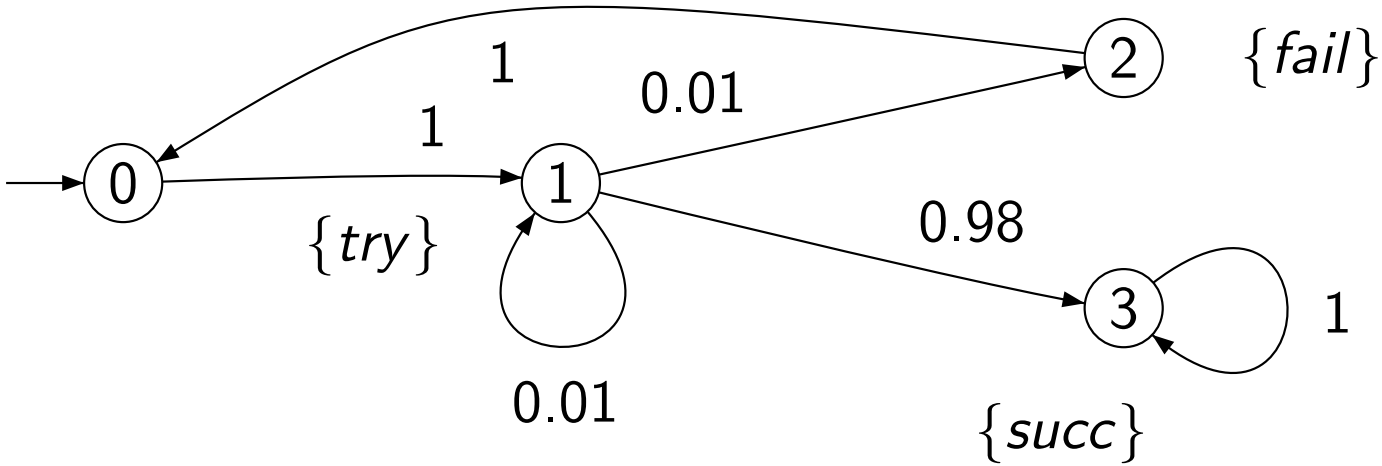
Beispiel 1



- Nach einem Zeitschritt: mit einer Wahrscheinlichkeit von 0.01 abwarten oder mit derselben scheitern oder mit der Wahrscheinlichkeit von 0.98 erfolgreich senden
- Scheitern: Prozess beginnt erneut

$$\mathcal{L}(1) = \{try\}, \mathcal{L}(2) = \{fail\}, \mathcal{L}(3) = \{succ\}$$

Beispiel 1



- Beispiel Kostenstruktur : $C = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$
- Häufigkeit, in der Zustand 1 begangen wird

Pfad: eine Ausführung eines Systems, daß von einer DTMC modelliert wird

- Formal: **Pfad**= nichtleere Zustandsfolge $s_0 s_1 s_2 \ldots$, in der $s_i \in S$ und $P(s_i, s_{i+1})$ für alle $i > 0$ ist.
- Pfad ist endlich oder unendlich
- $\omega(i)$: $i-$**ten Zustand** im Pfad ω,
- $|\omega|$: **Länge des** ω (Anzahl der Übergänge),
- ω_{fin}: **endlicher Pfad**,
- $last(\omega_{fin})$: **letzter Zustand** des Pfades,
- Wenn $\omega_{fin}(i) = \omega(i)$ für $0 < i \leq n$, dann ist endlicher Pfad ω_{fin} der Länge n **Präfix** des unendlichen Pfades ω.
- **Mengen aller endlichen bzw. unendlichen Pfade**: $Pfad_s^{fin}$ bzw. mit $Pfad_s$ gekennzeichnet.

Um das Verhalten der Wahrscheinlichkeiten vorherzusagen, definieren wir für jeden Zustand ein Wahrscheinlichkeitsmaß $Prob_s$ über der Menge $Pfad_s$.

- **Definition 1:** Für jeden endlichen Pfad $\omega_{fin} \in Pfad_s^{fin}$ ist die Wahrscheinlichkeit $P_s(\omega_{fin})$:

$$P_s(\omega_{fin}) = \begin{cases} 1 & \text{if } n = 0 \\ P(\omega(0), \omega(1)) \ldots P(\omega(n-1), \omega(n)) & \text{otherwise} \end{cases}$$

  wobei $n = |\omega_{fin}|$.

- **Definition 2:** Zylindermenge $C(\omega_{fin})$

$$C(\omega_{fin}) = \{\omega \in Path_s \mid \omega_{fin} \text{ ist Praefix von } \omega\}$$

  : Menge aller unendlichen Pfade mit Präfix ω_{fin}

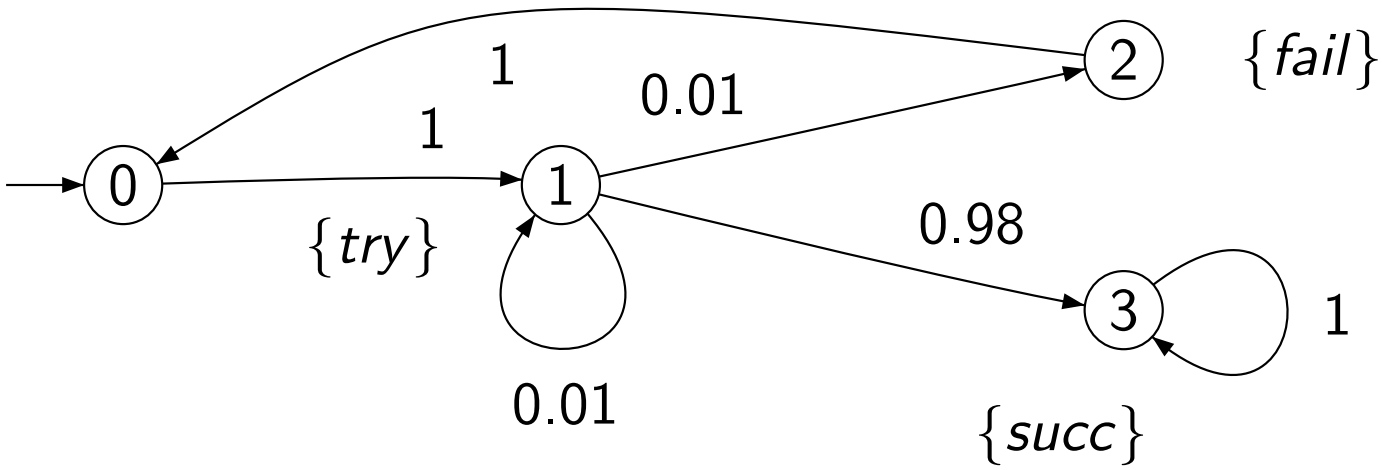
$Prob_s$

- **Definition 3:** Sei $\sum_s$ die kleinste $\sigma-$Algebra auf $Pfad_s$, die alle Mengen $C(\omega_{fin})$ enthält und ω_{fin} aus $Pfad_s^{fin}$. Wahrscheinlichkeitsmaß $Prob_s$:

$$Prob_s(C(\omega_{fin})) = P_s(\omega_{fin}) \qquad \text{für alle } \omega_{fin} \in Pfad_s^{fin}$$

- Durch Aufzeigen der Pfadmengen, die diese Anforderung erfüllen, kann die Wahrscheinlichkeit gemessen werden, dass sich eine DTMC auf eine genau bechriebene Weise verhält.

Beispiel 2



fünf verschiedene Pfade der Länge 3, die im Zustand 0 anfangen, das Wahrscheinlichkeitsmaß von den Zylindermengen dieser Pfade:

- $Prob_0(C(0111)) = 1 * 0.01 * 0.01 = 0.0001$
- $Prob_0(C(0112)) = 1 * 0.01 * 0.01 = 0.0001$
- $Prob_0(C(0113)) = 1 * 0.01 * 0.98 = 0.0098$
- $Prob_0(C(0133)) = 1 * 0.98 * 1 = 0.98$

- Erwartungswerte über Pfade definieren
- für gegebene Zielmenge von Zuständen F die erwarteten Kosten zum Erreichen eines Zustands in F erreichen.

Sei E_s die übliche Erwartung im Hinblick auf das Maß $Prob_s$ und für jeden Pfad $\omega \in Pfad_s$ gilt:
$cost(F)(\omega) =$

$$\begin{cases} \sum_{i=1}^{min\{\ j \mid \omega(j) \in F\ \}} c(\omega(i-1), \omega(i)) & \exists j \in \mathbb{N}\ mit\ \omega(j) \in F \\ \infty & sonst \end{cases}$$

- $E_s(cost(F))$= erwarteten Kosten F zu erreichen

PCTL

Probabilistic Computation Tree Logik(PCTL)

Spezifikationen von DTMCs schreiben mit Probabilistic Computation Tree Logic (PCTL).

Definition Syntax von PCTL :

- $\phi ::= \mathit{true} \mid a \mid \neg\phi \mid \phi \wedge \phi \mid \mathcal{P}_{\bowtie p}[\psi] \mid \mathcal{E}_{\bowtie c}[\phi]$,
- $\psi ::= \ \phi \mid \phi \mathcal{U}^{\leq k} \phi \mid \phi \mathcal{U} \phi$,

wobei a eine atomare Aussage ist, $\bowtie \in \{\leq, <, \geq, >\}$, $p \in [0, 1]$, $c \in \mathbb{R}^{\geq 0}$ und $k \in \mathbb{N}$

PCTL

Zustandsformeln ϕ und **Pfadformeln** ψ

- Zustandsformeln : um eine Eigenschaft eines DTMC anzugeben
- Pfadformeln : Parameter des Operators $\mathcal{P}_{\bowtie p}[\psi]$
- Ein Zustand s erfüllt $\mathcal{P}_{\bowtie p}[\psi]$, wenn die Wahrscheinlichkeit ein Pfad von s aus zu nehmen, der ψ erfüllt, im von $\bowtie$ p spezifizierten Intervall liegt.
- Dafür nutzen wir das Wahrscheinlichkeitsmaß über Mengen unendlicher Pfade(s.o.).

PCTL

Pfadvariablen : Operatoren der **Temporalen Logik** X(„next“), $\mathbb{U}^{\leq k}$ („boundet until“) und $\mathbb{U}$(„Until“). Zur Wiederholung:

- $X\phi$ ist wahr, wenn ϕ im nächsten Zustand erfüllt wird.
- $\phi_1 \mathbb{U}^{\leq k} \phi_2$ ist wahr, wenn ϕ_2 innerhalb von k Zeitschritten, und ϕ_1 vor diesem Zeitpunkt immer erfüllt ist.
- $\phi_1 \mathbb{U} \phi_2$ ist wahr, wenn ϕ_2 in einem zukünftigen unbestimmten Zeitpunkt, und ϕ_1 vor diesem Zeitpunkt immer erfüllt ist.
- $\mathcal{E}_{\bowtie c}[\phi]$ hat als Parameter eine Zustandsformel ϕ. Zustand erfüllt $\mathcal{E}_{\bowtie c}[\phi]$, wenn die erwarteten Kosten, um von s aus einen ϕ akzeptierenden Zustand zu erreichen, in $\bowtie c$

Semantik von PCTL

$s \models \phi$ für „s erfüllt ϕ „,
ω, der die Pfadformel ψ erfüllt $\omega \models \psi$. Semantik von PCTL :
Sei $\mathcal{D} = (S, \bar{s}, \mathbf{P}, L, \mathbf{C})$. Für jeden Zustand $s \in S$ ist die **Erfüllbarkeitsrelation** $\models$ definiert als:

- $s \models \mathit{true}$ für alle $s \in S$
- $s \models a \Leftrightarrow a \in L(s)$,
- $s \models \neg\phi \Leftrightarrow s \not\models \phi$,
- $s \models \phi_1 \wedge \phi_2 \Leftrightarrow s \models \phi_1 \wedge s \models \phi_2$,
- $s \models \mathcal{P}_{\bowtie p}[\psi] \Leftrightarrow p_s(\psi) \bowtie p$,
- $s \models \mathcal{E}_{\bowtie c}[\phi] \Leftrightarrow e_s(\phi) \bowtie c$, wobei

Semantik von PCTL

Aus der Basissyntax von PCTL weitere Operatoren ableiten:

- $\mathit{false} \equiv \neg \mathit{true}$
- $\phi_1 \vee \phi_2 \equiv \neg(\neg\phi_1 \wedge \neg\phi_2)$
- $\phi_1 \rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2$

- in Pfadformeln $\Diamond$ Operator erlaubt: wobei
  - $\Diamond\phi$(„diamond“): ϕ eventuell erfüllt
  - begrenzte Variante $\Diamond^{\leq k}\phi$: ϕ innerhalb von k Zeitschritten erfüllt
  - $\Diamond$ durch „until“ und „bounded until“ in PCTL:

$$\mathcal{P}_{\bowtie p}[\Diamond\phi] \equiv \mathcal{P}_{\bowtie p}[\mathit{true}\,\mathcal{U}\,\phi],$$
$$\mathcal{P}_{\bowtie p}[\Diamond^{\leq k}\phi] \equiv \mathcal{P}_{\bowtie p}[\mathit{true}\,\mathcal{U}^{\leq k}\,\phi]$$

PCTL

- $\Box$(„box“)erlaubt
  - $\Box\phi$: ϕ wahr in allen Zuständen
  - $\Box^{\leq k}\phi$: ϕ in den ersten k Schritten des Pfades wahr
  - Dies kann wiederrum mit $\Diamond$ beschrieben werden:

$$\Box\phi \equiv \neg\Diamond\neg\phi,$$
$$\Box^{\leq k}\phi \equiv \neg\Diamond^{\leq k}\neg\phi.$$

- keine Negation erlaubt für Pfadformeln

PCTL

Deshalb wird gezeigt: für einen Zustand s und Pfadformel ψ: $p_s(\neg\psi) = 1 - p_s(\psi)$, folgendes:

$$\begin{aligned}\mathcal{P}_{\geq p}[\Box\phi] &\Leftrightarrow p_s(\Box\phi) \geq p \\ &\Leftrightarrow 1 - p_s(\Diamond\neg\phi) \geq p \\ &\Leftrightarrow p_s(\Diamond\neg\phi) \leq 1 - p \\ &\Leftrightarrow \mathcal{P}_{\leq 1-p}[\Diamond\neg\phi].\end{aligned}$$

folgende Äquivalenzen:

$$\mathcal{P}_{\bowtie p}[\Box\phi] \equiv \mathcal{P}_{\neg\bowtie 1-p}[\Diamond\neg\phi],\ \mathcal{P}_{\bowtie p}[\Box^{\leq k}\phi] \equiv \mathcal{P}_{\bar{\bowtie} 1-p}[\Diamond^{\leq k}\neg\phi],$$

wobei $\bar{\leq} \equiv \geq,\ \bar{<} \equiv >,\ \bar{\geq} \equiv \leq$ *und* $\bar{>} \equiv <$

PCTL

Verwandtschaft zwischen der PCTL und der Temporalen Logik :

- $\exists \Diamond \phi \equiv \mathcal{P}_{>0}[\Diamond \phi]$

Da die Wahrscheinlichkeit nur größer null sein kann, wenn ein endlicher zu ϕ führender Pfad existiert.

- $\forall \Diamond \phi \not\equiv \mathcal{P}_{\geq 1}[\Diamond \phi]$

Umkehrung nicht äquivalent.

Beispiel 3

- $\mathcal{P}_{<0.1}[\Diamond^{\leq 10} error]$: Wahrscheinlichkeit, dass System innerhalb von 10 Zeitschritten einen Fehlerzustand erreicht ist kleiner als 0.1;
- $\mathcal{P}_{\geq 0.9}[\neg down \mathcal{U} served]$ Wahrscheinlichkeit, dass System nicht abbricht bis die Anfrage bedient wurde, ist größer als 0.9;
- $\mathcal{E}_{<10}[finished]$ die erwartete Anzahl verlorener Kunden vor Fertigstellung des Produkts ist 10;
- $\mathcal{E}_{\geq 14}[done]$ die erwartete Anzahl korrekt gesendeter Mitteilungen ist mindestens 14;

vermeintlich **Schwäche** von PCTL: es ist nicht möglich ist die eigentliche Wahrscheinlichkeit zu bestimmen, nur ob die Wahrscheinlichkeit eine spezielle Grenze hat. $\Rightarrow$ sichert aber, dass jede PCTL Formel boolischen Wert annimmt.

PCTL

- Praxis Einschränkung lockern: Ist äußerster Operator $\mathcal{P}_{\bowtie p}$ oder $\mathcal{E}_{\bowtie c}$, kann die Grenze $\bowtie p$ oder $\bowtie c$ weggelassen und einfach die Wahrscheinlichkeit bzw. die erwarteten Kosten berechnen.
- PCTL Modellprüfungsalgorithmus entwickelt sich durch die Berechnung der eigentlichen Wahrscheinlichkeit und anschließenden Vergleich mit der Grenze: keine zusätzliche Arbeit

PCTL Modellprüfung

PCTL Modellprüfung

Modellprüfungsalgorithmus für PCTL über DTMC

- **Eingabe**: gekennzeichnete DTMC $(S, \bar{s}, \mathcal{P}, \mathcal{L}, C)$ und die PCTL Formel ϕ.
- **Ausgabe**: Menge $Sat(\phi) = \{s \in S | s \models \phi\}$, die alle Zustände des Modells, dass ϕ erfüllt enthält.
- Der Algorithmus überprüft, ob jeder Zustand aus S erfüllt ist.

PCTL Modellprüfung

Analysebaum der Formel ϕ.

- Jeder Knoten des Baums: mit einer Subformel von ϕ gekennzeichnet.
- Blätter: *true* oder mit atomaren Ausdruck *a* gekennzeichnet.
- Während wir uns auf die Wurzel des Baums hocharbeiten, berechnen wir rekursiv die Menge der Zustände, die jede Subformel erfüllt.
- Am Ende haben wir bestimmt, ob ein Zustand im Modell ϕ erfüllt.

PCTL Modellprüfung

Der Algorithmus für die PCTL Operatoren :

$$
\begin{aligned}
Sat(true) &= S, \\
Sat(false) &= \emptyset, \\
Sat(a) &= \{s \mid a \in \mathcal{L}(s)\}, \\
Sat(\neg\phi) &= S \setminus Sat(\phi), \\
Sat(\phi_1 \wedge \phi_2) &= Sat(\phi_1) \cap Sat(\phi_2), \\
Sat(\phi_1 \vee \phi_2) &= Sat(\phi_1) \cup Sat(\phi_2), \\
Sat(\phi_1 \Rightarrow \phi_2) &= (S \setminus Sat(\phi_1)) \cup Sat(\phi_2), \\
Sat(\mathcal{P}_{\bowtie p}[\psi]) &= \{s \in S | p_s(\psi) \bowtie p\}, \\
Sat(\mathcal{E}_{\bowtie c}[\phi[) &= \{s \in S | e_s(\phi) \bowtie c\}.
\end{aligned}
$$

PCTL Modellprüfung

- Modellprüfung leicht zu implementieren mit Ausnahme von den $\mathcal{P}_{\bowtie p}[\psi]$ und $\mathcal{E}_{\bowtie c}[\phi[$ Operatoren
- Für diese : für jeden Zustand der DTMC die Wahrscheinlichkeit $p_s(\psi)$ und die erwarteten Kosten $e_s(\phi)$ berechnen und diese Werte mit der Grenze in der Formel vergleichen
- Werte für diese vier Fälle berechnen : $\mathcal{P}_{\bowtie p}[\phi]$, $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$, $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ und $\mathcal{E}_{\bowtie c}[\phi]$.

Der $\mathcal{P}_{\bowtie p}[\phi]$ Operator

Der $\mathcal{P}_{\bowtie p}[\phi]$ Operator

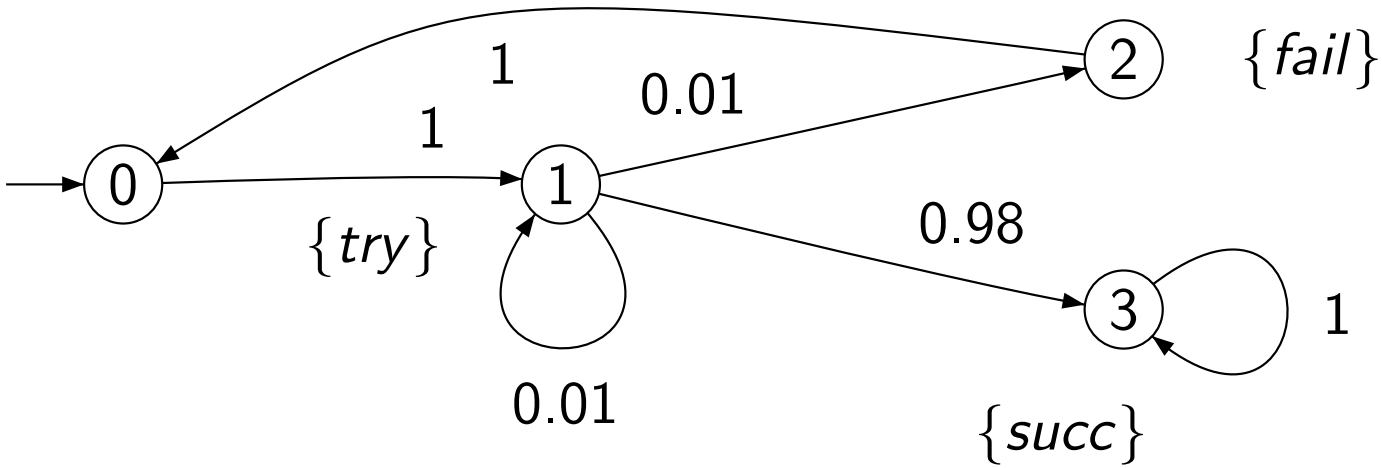
Wahrscheinlichkeit $p_s(\phi)$ für jeden Zustand $s \in S$ berechnen. Dies verlangt nur die Wahrscheinlichkeiten von **direkten Übergängen** von s aus :

$$p_s(\phi) = \sum_{s' \in Sat(\phi)} P(s, s')$$

Wahrscheinlichkeitsvektor $p(\phi)$ für alle Zustände: Sei der Spaltenvektor ϕ mit $\phi(s) = 1$ wenn $s \models \phi$, sonst 0, so wird $p(\phi)$ durch die Verwendung einer Matrixmultiplikation berechnet:

$$p(\phi) = P * \phi.$$

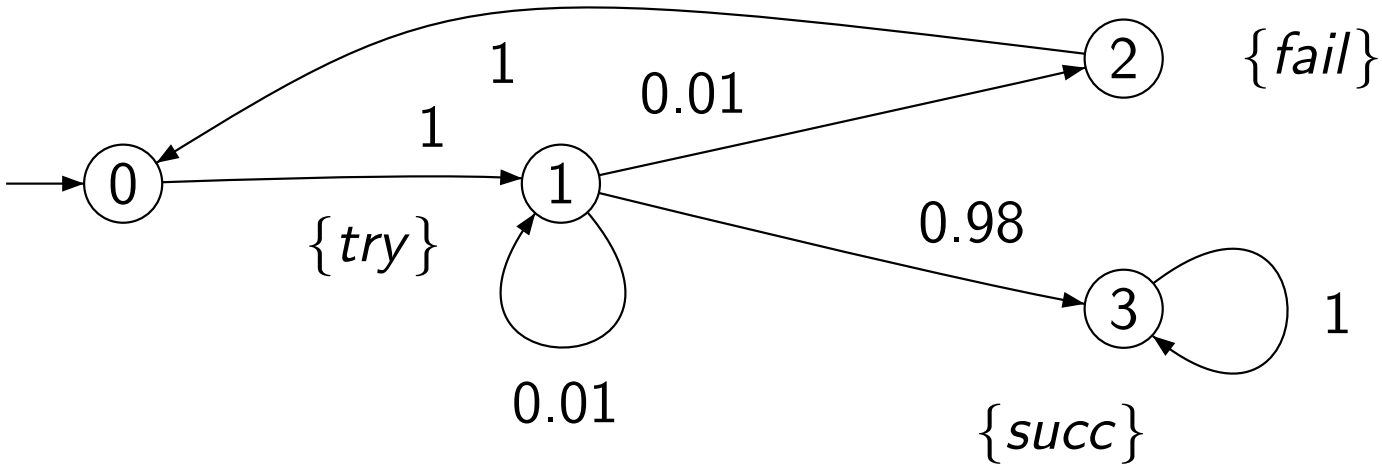
Beispiel zum $\mathcal{P}_{\bowtie p}[\phi]$ Operator



PCTL Formel $\mathcal{P}_{\geq 0.9}[X(\neg try \vee succ)]$. Rekursiv vorranschreitend aus der innersten Subformel berechnen wir:

$$
\begin{aligned}
Sat(try) &= \{1\}, \\
Sat(\neg try) &= S \setminus Sat(try) = \{0, 2, 3\}, \\
Sat(succ) &= \{3\}, \\
Sat(\neg try \vee succ) &= Sat(\neg try) \cup Sat(succ) = \{0, 2, 3\}.
\end{aligned}
$$

Beispiel zum $\mathcal{P}_{\bowtie p}[\phi]$ Operator



Dies hinterlässt den äußersten X Operator. Nehmen wir $\mathbf{P}$ wie oben an und definieren $\phi = [1, 0, 1, 1]$, dann :

$$p(\phi) = \mathbf{P} * \phi = [0, 0.99, 1, 1].$$

Deshalb ist $Sat(\mathcal{P}_{\geq 0.9}[X(\neg try \vee succ)]) = \{1, 2, 3\}$.

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ Operator

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ Operator

Wahrscheinlichkeiten $p_s[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ für alle $s \in S$ berechnen.
Menge S der Zustände in drei disjunkte Mengen, $S^{no}, S^{yes}, S^?$ teilen, wobei:

$$S^{no} = S \setminus (Sat(\phi_1) \cup Sat(\phi_2)),\ S^{yes} = Sat(\phi_2),$$
$$S^? = S \setminus (S^{no} \cup S_{yes}).$$

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ Operator

Wenn wir die Matrix P' definieren als :

$$P'(s,s') = \begin{cases} P(s,s') & \textit{wenn } s \in S^? \\ 1 & \textit{wenn } s \in S^{yes} \textit{ und } s = s' \\ 0 & \textit{sonst} \end{cases}$$

kann die verlangte Berechnung für alle Zustände so ausgedrückt werden:

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ Operator

Für die Zustände $S^?$ definieren wir die Wahrscheinlichkeit $p_s(\phi_1 \mathcal{U}^{\leq k} \phi_2)$: Für $k = 0$

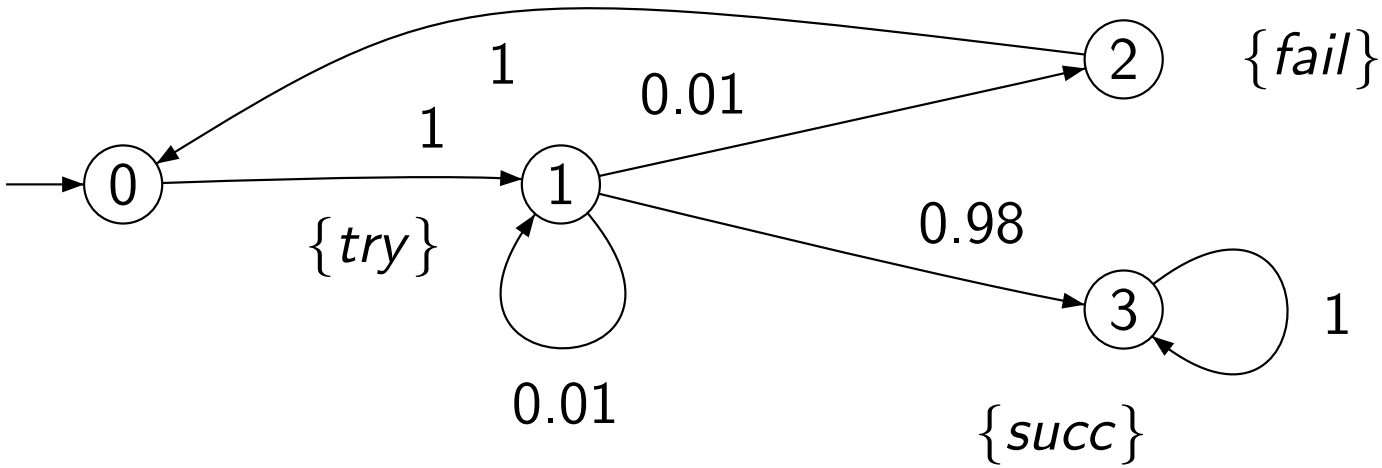
$$p_s(\phi_1 \mathcal{U}^{\leq 0} \phi_2)(s) = 1 \textit{ wenn } s \in S^{yes} \textit{ und } 0 \textit{ sonst}$$

Für $k > 0$

$$p_s(\phi_1 \mathcal{U}^{\leq k} \phi_2) = P' * p(\phi_1 \mathcal{U}^{\leq k-1} \phi_2)$$

k Matrixmultiplikationen erforderlich

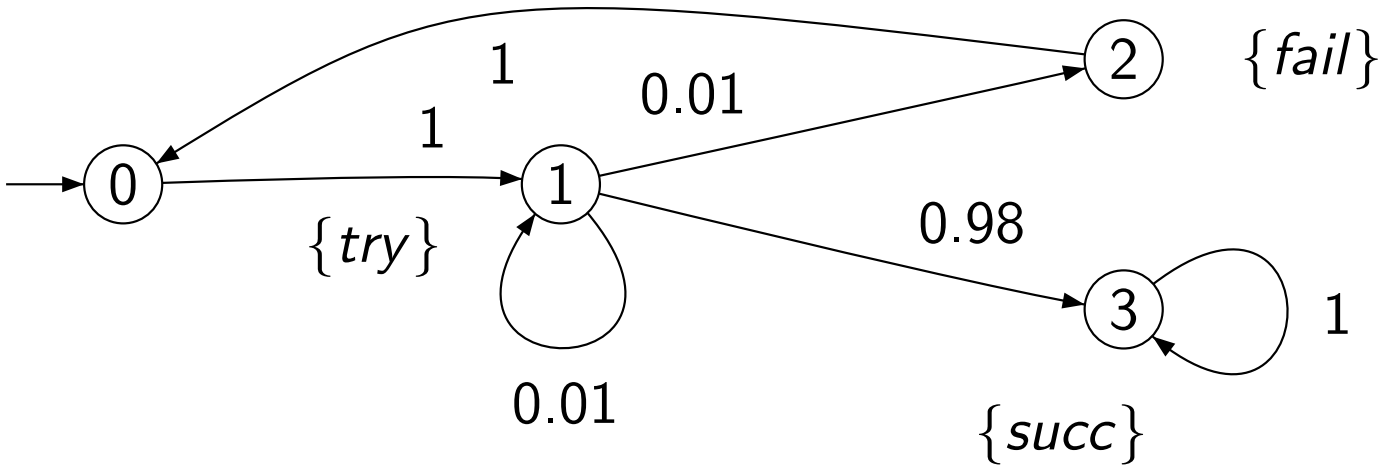
Beispiel zum $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ Operator



DTMC und PCTL Formel $\mathcal{P}_{>0.98}[\Diamond^{\leq 2} succ]$ äquivalent zu Formel $\mathcal{P}_{>0.98}[true \mathcal{U}^{\leq 2} succ]$.

$$Sat(true) = \{0, 1, 2, 3\},$$
$$Sat(succ) = \{3\},$$
$$S^{no} = S \setminus (Sat(true) \cup Sat(succ)) = \emptyset,$$
$$S^{yes} = Sat(succ) = \{3\}.$$

Beispiel zum $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U}^{\leq k} \phi_2]$ Operator



Matrix P' in diesem Fall identisch mit P. Wir berechnen:

$$p(\phi_1 \mathcal{U}^{\leq 0} \phi_2) = [0, 0, 0, 1],$$
$$p(\phi_1 \mathcal{U}^{\leq 1} \phi_2) = P' * [0, 0, 0, 1] = [0, 0.98, 0, 1],$$
$$p(\phi_1 \mathcal{U}^{\leq 2} \phi_2) = P' * [0, 0.98, 0, 1] = [0.98, 0.9898, 0, 1].$$

Deshalb ist $Sat(\mathcal{P}_{>0.98}[\diamond^{\leq 2} succ]) = \{1, 3\}$.

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ Operator

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ Operator

- für alle Zustände s $p_s(\phi_1 \mathcal{U} \phi_2)$ berechnen $\Rightarrow$ mit $Prob0$ bzw $Prob1$ Algorihmus :

$$S^{no} = Prob0(Sat(\phi_1), Sat(\phi_2)),$$
$$S^{yes} = Prob1((Sat\phi_1), Sat(\phi_2), S^{no}),$$
$$S^{?} = S \setminus (S^{no} \cup S^{yes}).$$

Der *Prob*0 Algorithmus

function $\textsc{Prob0}(Sat(\phi_1), Sat(\phi_2))$
$R := Sat(\phi_2)$
$done := false$
while $done = false$ **do**
  $R' := R \cup \{s \in Sat(\phi_1) | \exists s' \in R.P(s, s') > 0\}$
  if $R' = R$ **then**
    $done := true$
  $R := R'$
return $S \setminus R$
end function

Der *Prob*1 Algorithmus

function PROB1($Sat(\phi_1), Sat(\phi_2), S^{no}$)
$R := S^{no}$
$done := false$
while $done = false$ **do**
  $R' := R \cup \{s \in (Sat(\phi_1) \setminus Sat(\phi_2)) | \exists s' \in R.P(s, s') > 0\}$
  if $R' = R$ **then**
    $done := true$
  $R := R'$
return $S \setminus R$
end function

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ Operator

Wahrscheinlichkeiten $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ für alle Zustände können nur durch Lösen des Linearen Gleichungssytems nach den Variablen $\{x_s | s \in S\}$ berechnet werden:

$$x_s = \begin{cases} 0 & \textit{wenn } s \in S^{no} \\ 1 & \textit{wenn } s \in S^{yes} \\ \sum_{s' \in S} P(s, s') * x_{s'} & \textit{wenn } s \in S^{?} \end{cases}$$

und dann $p_s(\phi_1 \mathcal{U} \phi_2) = x$.Hier ist P' hierdurch gegeben:

$$P'(s, s') = \begin{cases} P(s, s') & \textit{wenn } s \in S^{?} \\ 0 & \textit{sonst} \end{cases}$$

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ Operator

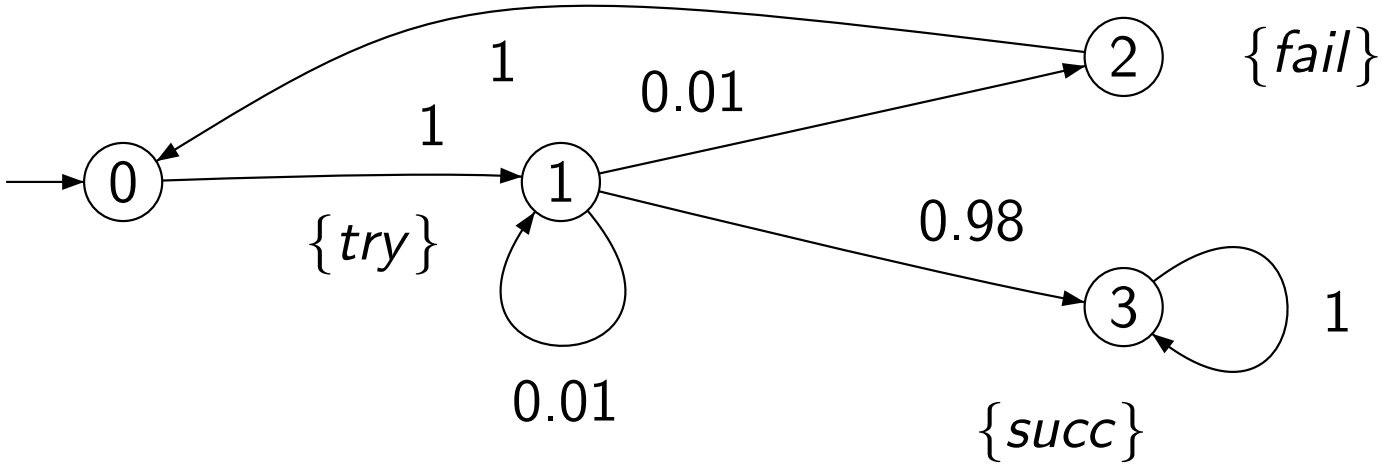
$$x_s = \sum_{s' \in S^?} P(s, s') * x_{s'} + \sum_{s' \in S^{yes}} P(s, s')$$

Da wir die Probleme für sehr große Modelle lösen wollen, eignen sich iterative Methoden gut.

Der $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ Operator

- *Prob*0 und *Prob*1 ist der erste Teil der Berechnung von $p_s(\phi_1 \mathcal{U} \phi_2)$. Deshalb sprechen wir hier von **Vorberechnungsalgorithmen**.

Beispiel zum $\mathcal{P}_{\bowtie p}[\phi_1 \mathcal{U} \phi_2]$ Operator



PCTL Formel $\mathcal{P}_{>0.99}[try \mathcal{U} succ]$: $Sat(try) = \{1\}$ und $Sat(succ) = \{3\}$. *Prob*0 bestimmt in zwei Iterationen $S^{no} = \{0, 2\}$. *Prob*1 bestimmt $S^{yes} = \{3\}$. Entstehendes LGS :

$$x_0 = 0, \quad x_1 = 0.01x_1 + 0.01x_2 + 0.98x_3, \quad x_2 = 0, \quad x_3 = 1.$$

Das führt zu der Lösung $(x_0, x_1, x_2, x_3) = (0, 98/99, 0, 1)$. Setze $p_s(try \mathcal{U} succ) = x_s$. Die Formel $\mathcal{P}_{>0.99}[try \mathcal{U} succ]$ kann nur im Zustand 3 erfüllt sein.

Der $\mathcal{E}_{\bowtie c}[\phi]$ Operator

Der $\mathcal{E}_{\bowtie c}[\phi]$ Operator

- für alle Zustände die erwarteten Kosten $e_s(\phi)$ berechnen
- Zustandsmengen identifizieren, für die $e_s(\phi)$ 0 oder ∞ ist, (S^0 oder S^∞).
- S^0 ist aus den Zuständen, die ϕ erfüllen
- S^∞ aus den Zuständen ist, die eine kleinere Wahrscheinlichkeit als 1 haben einen ϕ Zustand zu erreichen. Berechne $Prob0$ und $Prob1$:

$$S^0 = Sat(\phi),$$
$$S^\infty = S \setminus Prob1(S, Sat(\phi), Prob0(S, Sat(\phi))),$$
$$S^? = S \setminus (S^0 \cup S^\infty).$$

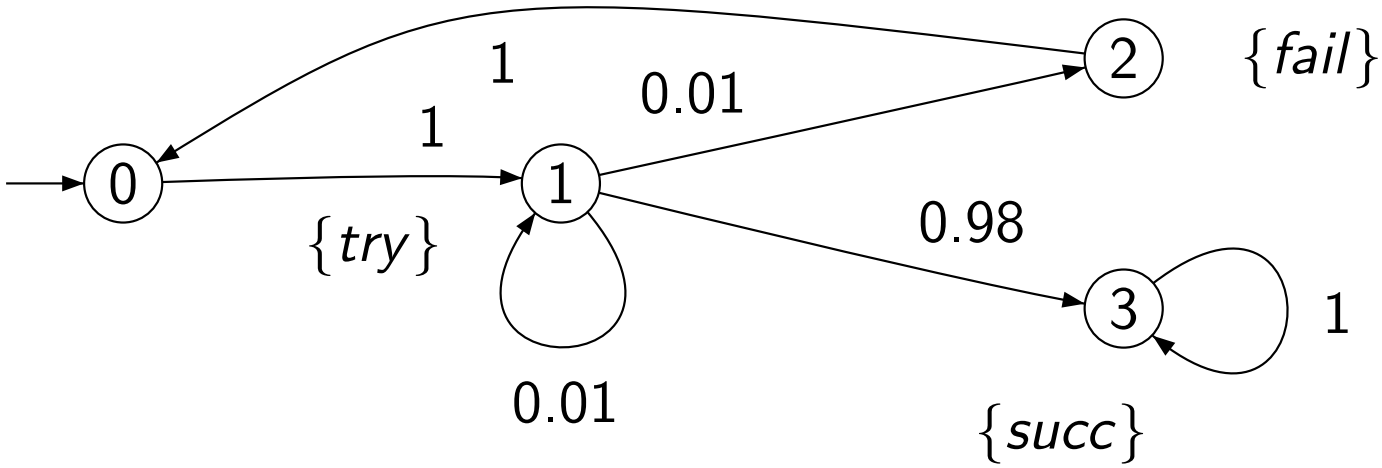
Der $\mathcal{E}_{\bowtie c}[\phi]$ Operator

Wir müssen trotzdem $e_s(\phi)$ für die verbleibenden Zustände $S^?$ ausrechnen. Dies wird durch Lösen des LGS nach den Variablen $[x_s | s \in S^?]$ getan :

$$x_s = \sum_{s' \in S^?} P(s, s').(C(s, s') + x_{s'})$$

und anschließend $e_s = x_s$ für $s \in S^?$ ausrechnen.

Beispiel zum $\mathcal{E}_{\bowtie c}[\phi]$ Operator



Wir hatten jedem Übergang, der Zustand 1 betrat, die Kosten 1, dem Rest 0 zugeordnet. Dann sichert die PCTL Formel $\mathcal{E}_{<2}\{succ\}$ zu, dass die Häufigkeit, an der in Zustand 1 begangngen wird bevor ein Zustand *succ* erfüllt ist, kleiner 2 ist. Prozedur berechnen:

$$S^0 = Sat(succ) = \{3\},$$
$$Prob0(S, Sat(\phi)) = \emptyset$$
$$Prob1(S, Sat(\phi), \emptyset) = \{0, 1, 2, 3\},$$

Beispiel zum $\mathcal{E}_{\bowtie c}[\phi]$ Operator

$$S^{\infty} = S \setminus \{0,1,2,3\} = \emptyset,$$
$$S^{?} = S \setminus \{3\} = \{0,1,2\}.$$

führt zum LGS:
$x_0 = 1*(1+x_1),\ \ x_1 = 0.01*(1+x_1)+0.01*(0+x_2),\ \ x_2 = 1*(0+x_0),$
was zu der Lösung führt:
$(x_0, x_1, x_2) = (198/98, 100/98, 198/98)$.
Deshalb ist der Vektor der erwarteten Kosten
$e(succ) = (198/98, 100/98, 198/98, 0)$ und
$Sat(\mathcal{E}_{<2}[succ]) = \{1,3\}$.

Die Komplexität der PCTL Modellprüfung

- Zeitkomplexität für die Modellprüfung einer PCTL Formel ϕ ist in $|\phi|$ und polynomial in $|S|$
- Die Modellprüfung muss für jedes der $|\phi|$ Operatoren ausgeführt werden und jeder Operator ist im schlechtesten Fall polynomial in $|S|$.
- teuersten Operatoren : „until“ und „erwartete Kosten“ Operatoren, für welche ein LGS der Größe $|S|$ gelöst werden muss.

Fallstudie: das IPv4 ZeroConf Protokoll

Fallstudie: das IPv4 ZeroConf Protokoll

- Beschreibung einer Fallstudie: Das dynamische Konfigurationsprotokoll für IPv4(Internet Protokoll Version 4, die erste Version, die weltweit verbreitet war) verbindungslokaler Adressen⇒ Nützlichkeit der PCTL Modellprüfung
- abstraktes DTCM Modell und zeigen wie die PCTL für die Analyse gebraucht werden kann.

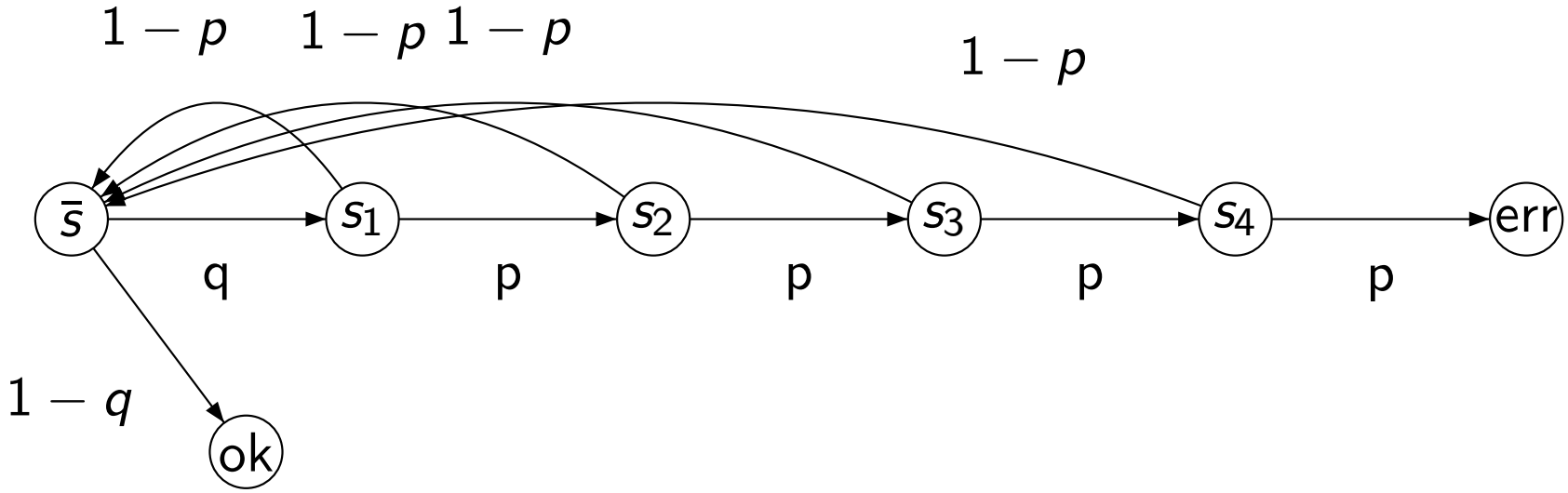
Fallstudie: das IPv4 ZeroConf Protokoll

- Beim Verbinden zum Netzwerk ist ein Gerät(host) notwendig, um eine IP Adresse von einer Datenbasis von 65024 möglichen Adressen zufällig auszuwählen.
- Internet Assigned Number Authority(IANA) hat die IP Adressen 169.254.1.0 – 169.254.254.255 genau für diesen Vorschlag allokiert.
- Der Host sendet vier ARP(Adress Resolution Protokoll) Packete, genannt Proben, zu allen anderen Hosts im Netzwerk. Probes: beinhalten die vom Host gewählte IP Adresse, verfassen Anfragen, um die Adresse zu nutzen und werden an zwei weitere Bereiche gesendet.
- ein anderer Host bereits diese Adresse im Gebrauch, muss er mit einem ARP Packet antworten, um seine Adressenanfrage zu sichern. Originalhost muss Protokoll neu starten, eine neue IP Adresse auswählen und neue Probes schicken.

Fallstudie: das IPv4 ZeroConf Protokoll

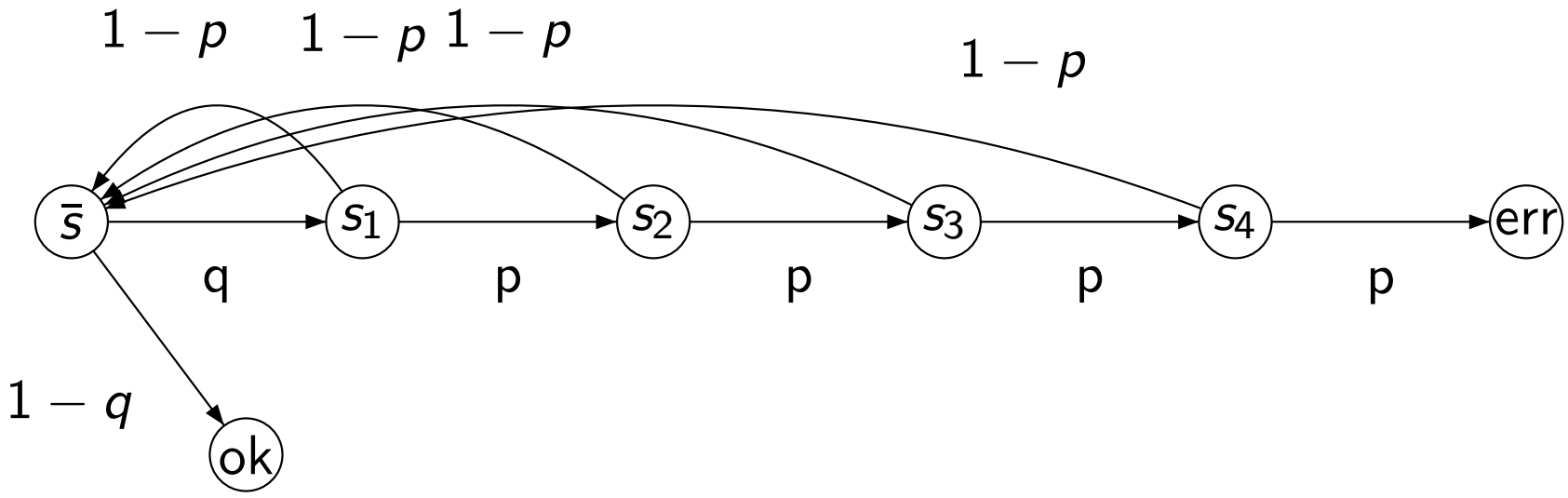
- Verhalten eines einzelnen Hosts, der vesucht eine IP Adresse zu konfigurieren in einem Netzwerk, in dem M hosts bereits verbunden sind.
- es gibt 65024 Adressen $\rightarrow$ Wahrscheinlichkeit, dass die zufällig gewählte IP Adresse bereits verwendet wird $q = M/65024$.
- Wir nehmen an der Host schickt N Proben und, dass mit der Wahrscheinlichkeit p keine Antwort kommt, wenn der Host Probes mit einer bereits genutzten IP Adresse schickt.
- Die Wahrscheinlichkeit stellt die Möglichkeit dar, dass die Probe verloren gegangen ist, der andere Host beschäftigt ist oder die Antwort verloren geht.

Fallstudie: das IPv4 ZeroConf Protokoll



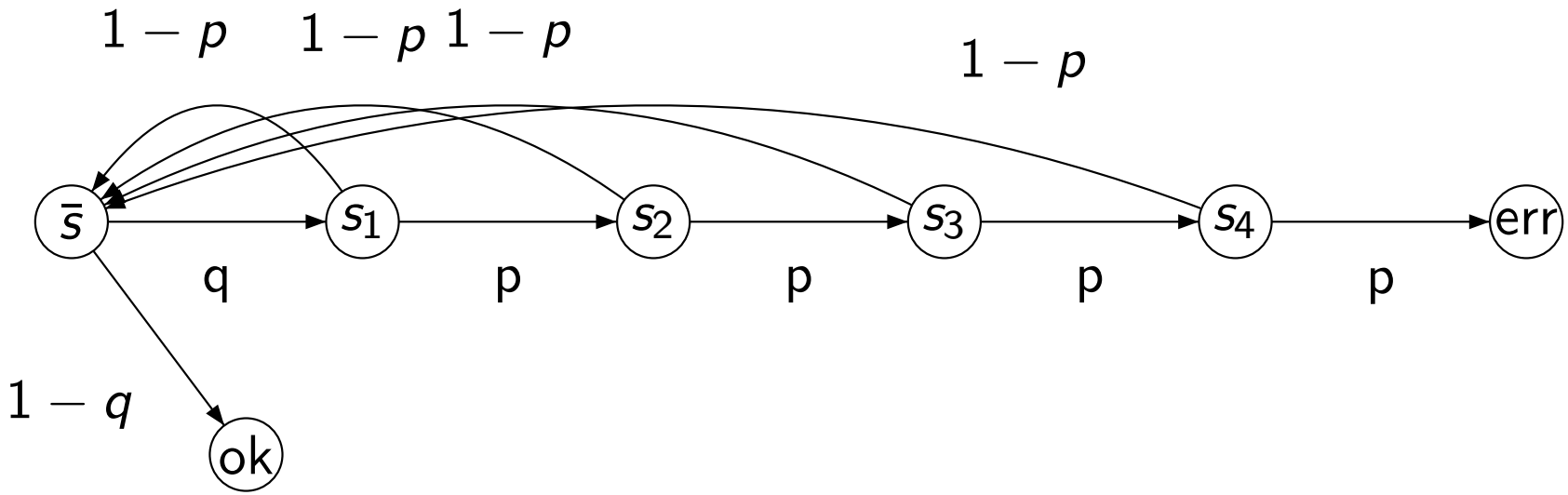
- DTMC hat $N + 3$ Zustände: $\{\bar{s}, s_1, ..., s_N, ok, err\}$.
- Im Anfangszustand $\bar{s}$ wählt der Host zufällig eine IP Adresse;
- In den Zusänden $\{s_i | 1 \leq i \leq N\}$ sendet der Host seine $i$te Probe; Im Zustand *ok* hat der Host erfolgreich eine „frische“ IP Adresse gewählt; Und im Zustand *err* ist die gewählte Adresse bereits im Gebrauch von einem anderen Host.

Fallstudie: das IPv4 ZeroConf Protokoll



- In $\bar{s}$: Mit der Wahrscheinlichkeit q gewählte IP Adresse bereits benutzt und der Host bewegt sich in den Zustand s_1;
- mit $1 - q$ ist die Adresse „frisch“ und der Host geht in den Zustand ok. Im Zustand s_i, mit $1 \leq i < N$ schickt der Host eine Probe.
- Mit p bekommt der Host keine Antwort und sendet eine andere Probe , während er in den Zustand s_{i+1} übergeht.

Fallstudie: das IPv4 ZeroConf Protokoll



- Mit $1 - p$ bekommt der Host eine Antwort und rekonfiguriert seine IP Adresse, indem er nach Zustand $\bar{s}$ übergeht.
- Zustand S_N : wenn der Host keine Antwort auf sein Probe bekommt, fängt er an eine bereits benutze IP Adresse zu nutzen und zum Zustand *err* überzugehen.

Fallstudie: das IPv4 ZeroConf Protokoll

Übergangswahrscheinlichkeitsmatrix einer DTMC im Allgemeinen:

$$P = \begin{pmatrix} 0 & q & 0 & 0 & \cdots & 0 & 1-q & 0 \\ 1-p & 0 & p & 0 & \cdots & 0 & 0 & 0 \\ 1-p & 0 & 0 & p & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 1-p & 0 & 0 & 0 & \cdots & p & 0 & 0 \\ 1-p & 0 & 0 & 0 & \cdots & 0 & 0 & p \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{pmatrix}$$

Wir assoziieren eine Kostenstruktur mit der DTMC, die der Zeit entspricht, die benötigt wird , um eine IP Adresse auszuwählen plus die zusätzlichen Kosten, die entstehen, wenn eine bereits genutze Adresse genommen wird.

Fallstudie: das IPv4 ZeroConf Protokoll

Host sendet im 2 Sekunden Intervall die Proben. Kostenstruktur:

- Übergänge aus den Zuständen $s_1 ..., s_{N-1}$: Kosten 2, da eine einzelne Probe in jedem dieser Zustände gesendet wird.
- Übergang von s_N zu $\bar{s}$: Kosten 2.
- Übergang von s_N nach err : R zugeteilt, da sobald der Übergang durchlaufen wird, der host eine bereits benutzte IP Adresse nutzt. Der Wert von R hängt davon ab, wie schädlich es für zwei hosts ist diesselbe IP Adresse zu nutzen, ws wiederum vom Netzwerk und der darin enthaltenen Geräte abhängt.
- Übergängen zwischen $\bar{s}$ und ok : Kosten $2N$. In diesem Zustand sendet der host N Proben im Zwei-Sekunden Intervall. Es wird keine Antworten zu den Proben geben, da die IP Adresse „frisch „ ist.
- Alle anderen Übergänge : Kosten null.

Fallstudie: das IPv4 ZeroConf Protokoll

Die Kostenstruktur des DTMC in Matrixform :

$$c = \begin{pmatrix} 0 & 0 & 0 & 0 & . & . & . & 0 & 2N & 0 \\ 2 & 0 & 2 & 0 & . & . & . & 0 & 0 & 0 \\ 2 & 0 & 0 & 2 & . & . & . & 0 & 0 & 0 \\ . & . & . & . & . & & & . & . & . \\ . & . & . & . & & . & & . & . & . \\ . & . & . & . & & & . & . & . & . \\ 2 & 0 & 0 & 0 & . & . & . & 2 & 0 & 0 \\ 2 & 0 & 0 & 0 & . & . & . & 0 & 0 & R \\ 0 & 0 & 0 & 0 & . & . & . & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & . & . & . & 0 & 0 & 0 \end{pmatrix}$$

Fallstudie: das IPv4 ZeroConf Protokoll

Sei a_s der atomaren Ausdruck, der nur im Zustand s wahr ist(z.B. $L(a_s) = \{s\}$). Betrachte die Eigenschaften :

- $\mathcal{P}_{\bowtie p}\{\Diamond a_{err}\}$- Die Wahrscheinlichkeit eine bereits verwendete IP Adresse zu nutzen ist $\bowtie p$;
- $\mathcal{E}_{\bowtie r}[a_{ok} \vee a_{err}]$- Die erwarteten Kosten, um eine IP Adresse zu finden sind $\bowtie r$.

Diese Eigenschaften werden durch Variieren der Modellparameter N(Anzahl gesendeter Proben) und p(Wahrscheinlichkeit Mitteilungen zu verlieren) ermittelt.

Fallstudie: das IPv4 ZeroConf Protokoll

Number of probes sent (N)	Probability of message loss (p)			
	0	0.001	0.01	0.01
1	0	1.56e-5	1.56e-4	0.001560
2	0	1.56e-8	1.56e-6	1.56e-4
3	0	1.56e-11	1.56e-8	1.56e-5
4	0	1.56e-14	1.56e-10	1.56e-6
5	0	1.56e-17	1.56e-12	1.56e-7
6	0	1.56e-20	1.56e-14	1.56e-8

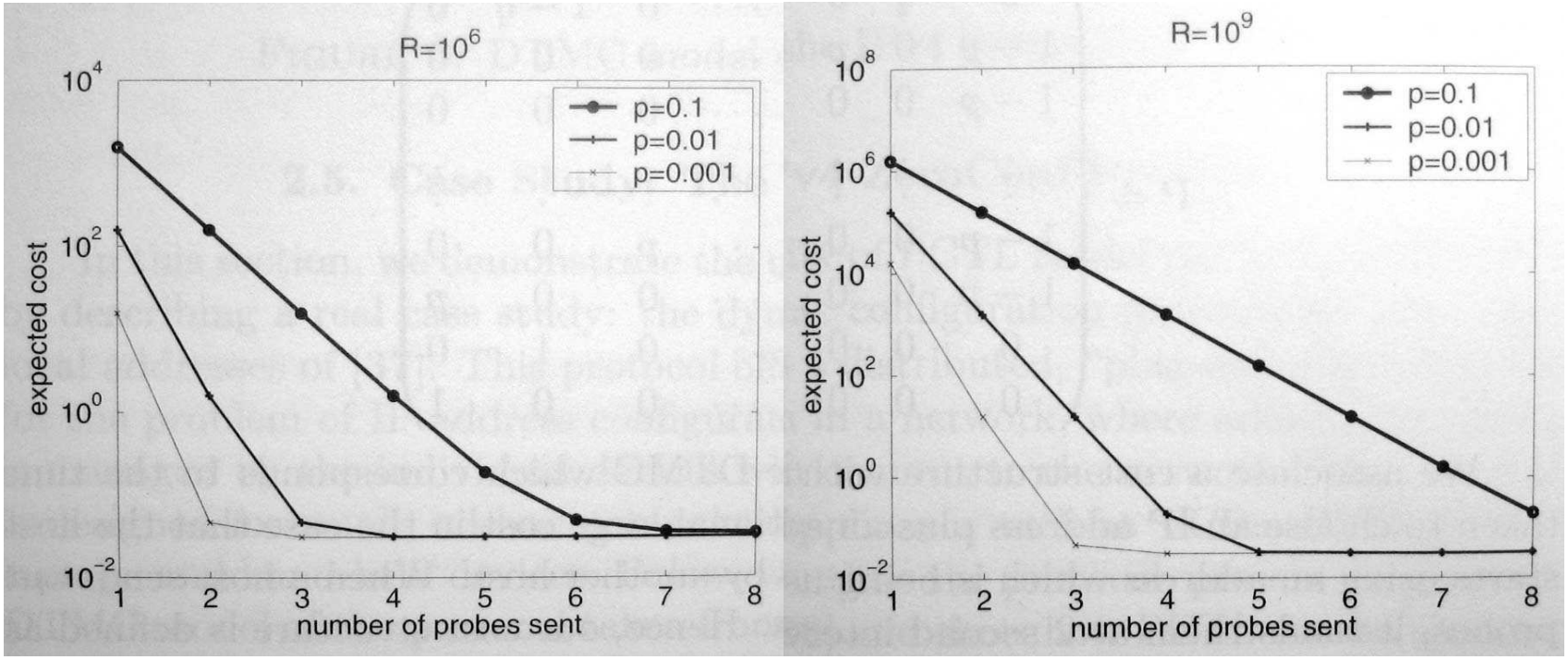
Hier wird von den erzielten Wahrscheinlicheiten für den Anfangszustand beim Prüfen der Eigenschaft $\mathcal{P}_{\bowtie p}[\Diamond a_{err}]$ berichtet. Die erzielten Ergebnisse sind wie erwartet.

Fallstudie: das IPv4 ZeroConf Protokoll

Number of probes sent (N)	Probability of message loss (p)			
	0	0.001	0.01	0.01
1	0	1.56e-5	1.56e-4	0.001560
2	0	1.56e-8	1.56e-6	1.56e-4
3	0	1.56e-11	1.56e-8	1.56e-5
4	0	1.56e-14	1.56e-10	1.56e-6
5	0	1.56e-17	1.56e-12	1.56e-7
6	0	1.56e-20	1.56e-14	1.56e-8

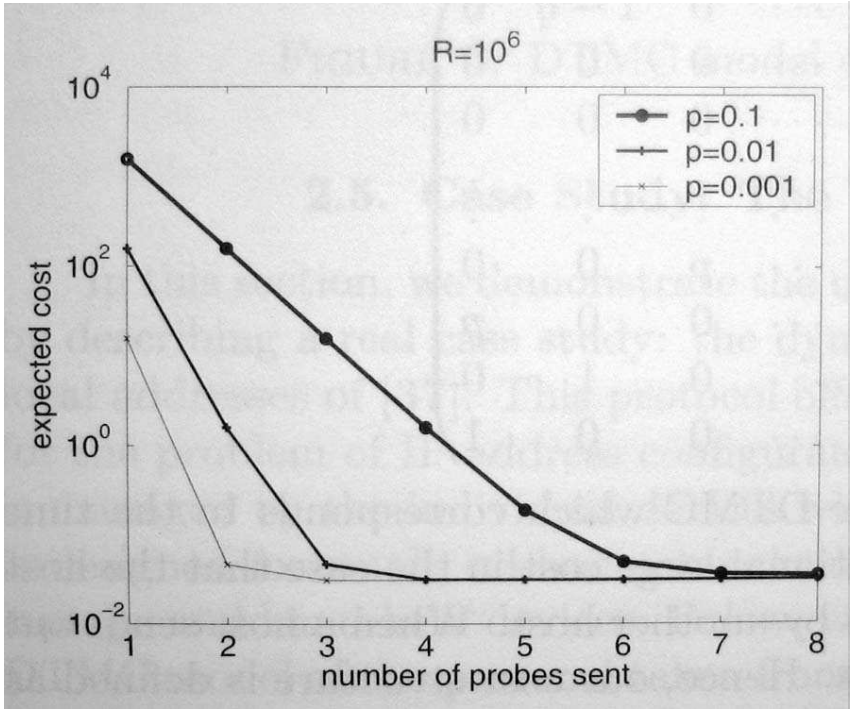
Die sinkende Wahrscheinlichkeit verlorener Proben senkt die Wahrscheinlichkeit von mehrfach genutzen IP Adressen. Der Anstieg der Anzahl gesendeter Proben senkt die Wahrscheinlichkeit von doppelt genutzen IP Adressen.

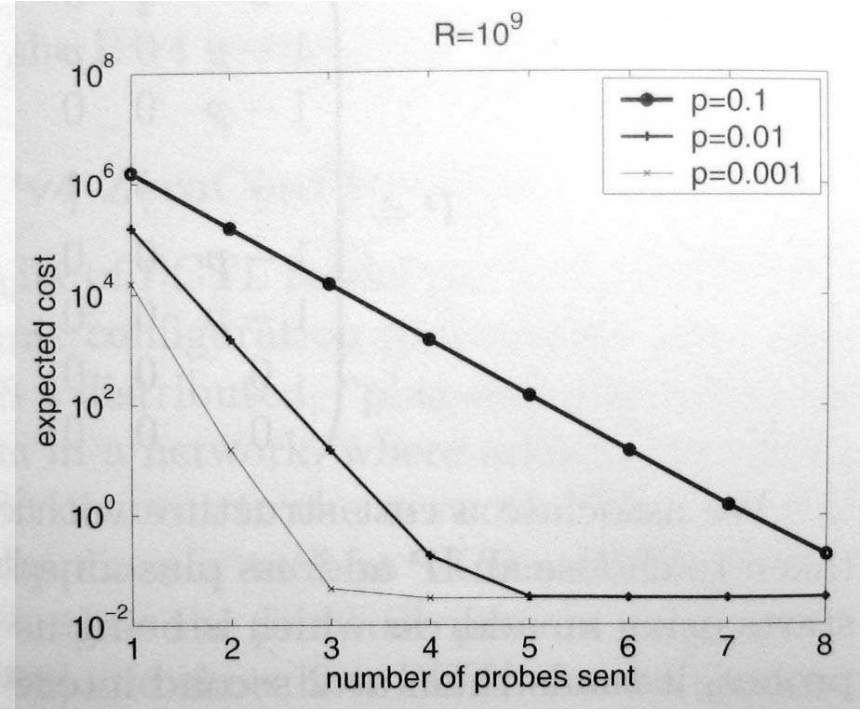
Fallstudie: das IPv4 ZeroConf Protokoll



Oben werden die erwarteten Kosten dargestellt erzielt bei der Eigenschaftsprüfung $\mathcal{E}_{\bowtie r}[a_{ok} \vee a_{err}]$ für die Fälle $R = 10^6$ und $R = 10^9$. Sowie die Wahrscheinlichkeit verlorener Nachrichten steigt, steigt auch die Anzahl gesendeter Proben, um die erwarteten Kosten zu reduzieren.

Fallstudie: das IPv4 ZeroConf Protokoll





Jedoch könnten die erwarteten Kosten wieder steigen, wenn zu viele Proben gesendet wurden. Obwohl die ansteigende Anzahl gesendeter Proben die Wahrscheinlichkeit doppelt genutzter Adressen senkt, steigert es die erwartete Zeit, um eine Adresse auszuwählen(da je mehr Proben man sendet, desto mehr Zeit braucht man).

Fallstudie: das IPv4 ZeroConf Protokoll

Modell kann erweitert werden, um die Wahrscheinlichkeit auszurechnen, dass ein Host eine „frische“ Adresse findet mit dem Senden von höchstens K Proben oder um die Wahrscheinlichkeit auszurechnen, wenn es Grenzen für die vergehende Zeit und Anzahl gesendeter Proben gibt.

Literaturverzeichnis


Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems *J.J.M.M.Rutten, Marta Kwiatkowska, Gethin Norman, David Parker*