

Technická diagnostika

Jiří Vomlel

**Laboratoř inteligentních systémů
Vysoká škola ekonomická Praha**

**Tato prezentace je k dispozici na:
<http://www.utia.cas.cz/vomlel/>**

Technická diagnostika - popis problému

- Příčiny problému (závady) $C \in \mathcal{C}$.
- Akce $A \in \mathcal{A}$ - opravné kroky, které mohou odstranit závadu.
- Otázky $Q \in \mathcal{Q}$ - kroky, které mohou pomoci identifikovat, kde je závada.
- Ke každé akci i otázce je přiřazena cena (c_A značí cenu akce A , c_Q cenu otázky Q). Cena může znamenat:
 - dobu potřebnou k provedení akce či otázky,
 - cenu za náhradní díl, který použijeme
 - rizikovost akce
 - nějaká kombinace výše uvedených.

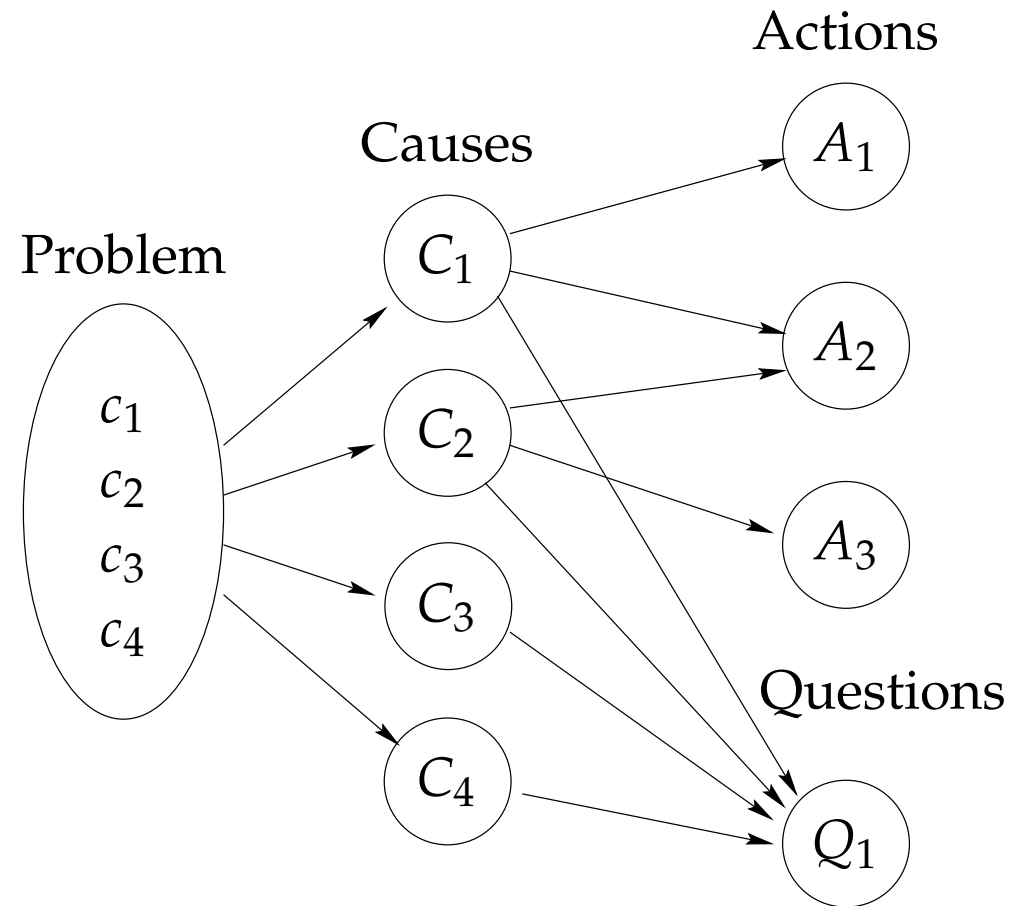
Příklad technické diagnostiky tiskárny

Trouble: světlý tisk.

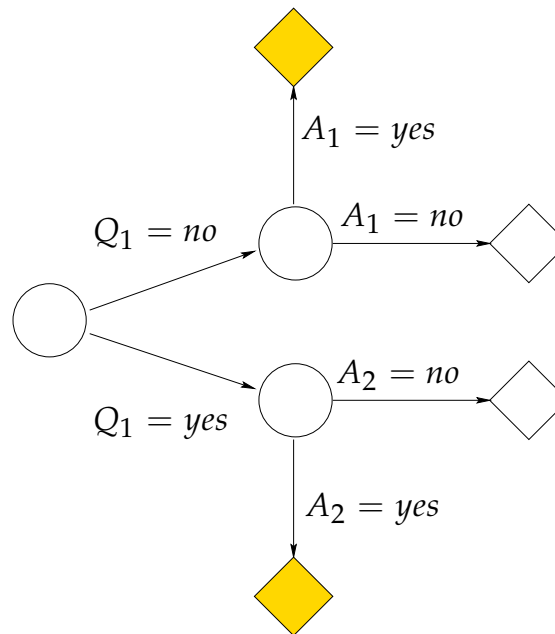
Troubleshooter: doporučí kroky, které pomohou odstranit “trouble”

Akce a otázky	cena
A_1 : Remove, shake and reseal toner	5
A_2 : Try another toner	15
A_3 : Cycle power	1
Q_1 : Is the printer configuration page printed light?	2
Možné závady při světlém tisku	$P(C_i)$
C_1 : Toner low	0.4
C_2 : Defective toner	0.3
C_3 : Corrupted dataflow	0.2
C_4 : Wrong driver setting	0.1

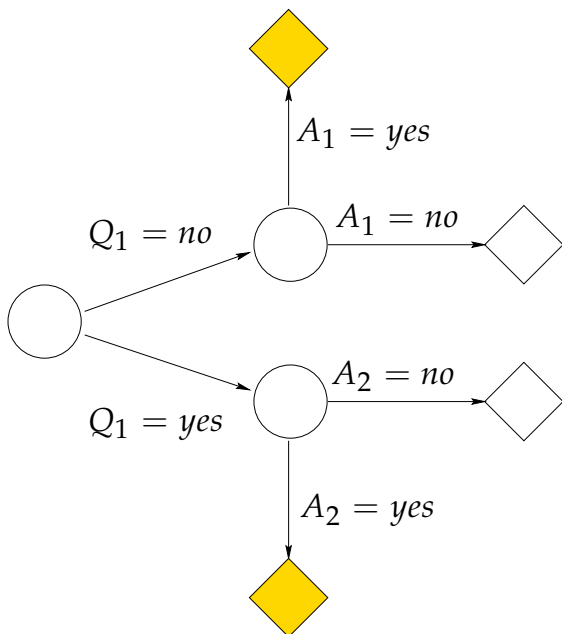
Light Print Problem - Bayesian Network



Světlý tisk - strategie odstranění závady



Světlý tisk - očekávaná cena opravy



Strategie	očekávaná cena opravy (ECR)
$Q_1 \left\{ \begin{array}{l} A_1 \\ A_2 \end{array} \right.$	$ \begin{aligned} & p(Q_1 = no, A_1 = yes) \cdot (c_{Q_1} + c_{A_1} + 0) \\ & + p(Q_1 = no, A_1 = no) \cdot (c_{Q_1} + c_{A_1} + c_{CS}) \\ & + p(Q_1 = yes, A_2 = yes) \cdot (c_{Q_1} + c_{A_2} + 0) \\ & + p(Q_1 = yes, A_2 = no) \cdot (c_{Q_1} + c_{A_2} + c_{CS}) \end{aligned} $

Očekávaná cena opravy (ECR)

Strategie může skončit

- neúspěšně (např. vyčerpali jsme všechny možné akce)
uplatní se penalizace $c(\mathbf{e}_\ell)$
penalta může být interpretována jako cena, kterou zaplatíme za zavolání servisních techniků
- úspěšně vyřešením problému $c(\mathbf{e}_\ell) \stackrel{\text{def}}{=} 0$

$$\begin{aligned}
\text{Node } n &\mapsto \mathbf{e}_n = \left\{ \begin{array}{l} (A = \text{yes/no})_{A \in \{\text{performed actions}\}} , \\ (Q = \text{yes/no})_{Q \in \{\text{performed questions}\}} \end{array} \right\} \\
&\mapsto p(\mathbf{e}_n) \dots \text{probability of getting to node } n \\
&\mapsto t(\mathbf{e}_n) \dots \text{total cost of actions and questions performed} \\
&\quad \text{(to get to node } n)
\end{aligned}$$

$$ECR(\mathbf{s}) = \sum_{\ell \in \{\text{terminal nodes of } \mathbf{s}\}} p(\mathbf{e}_\ell) \cdot [t(\mathbf{e}_\ell) + c(\mathbf{e}_\ell)]$$

Cíl technické diagnostiky

Cíl: nalézt strategii s^* takovou, že pro všechny přípustné strategie s

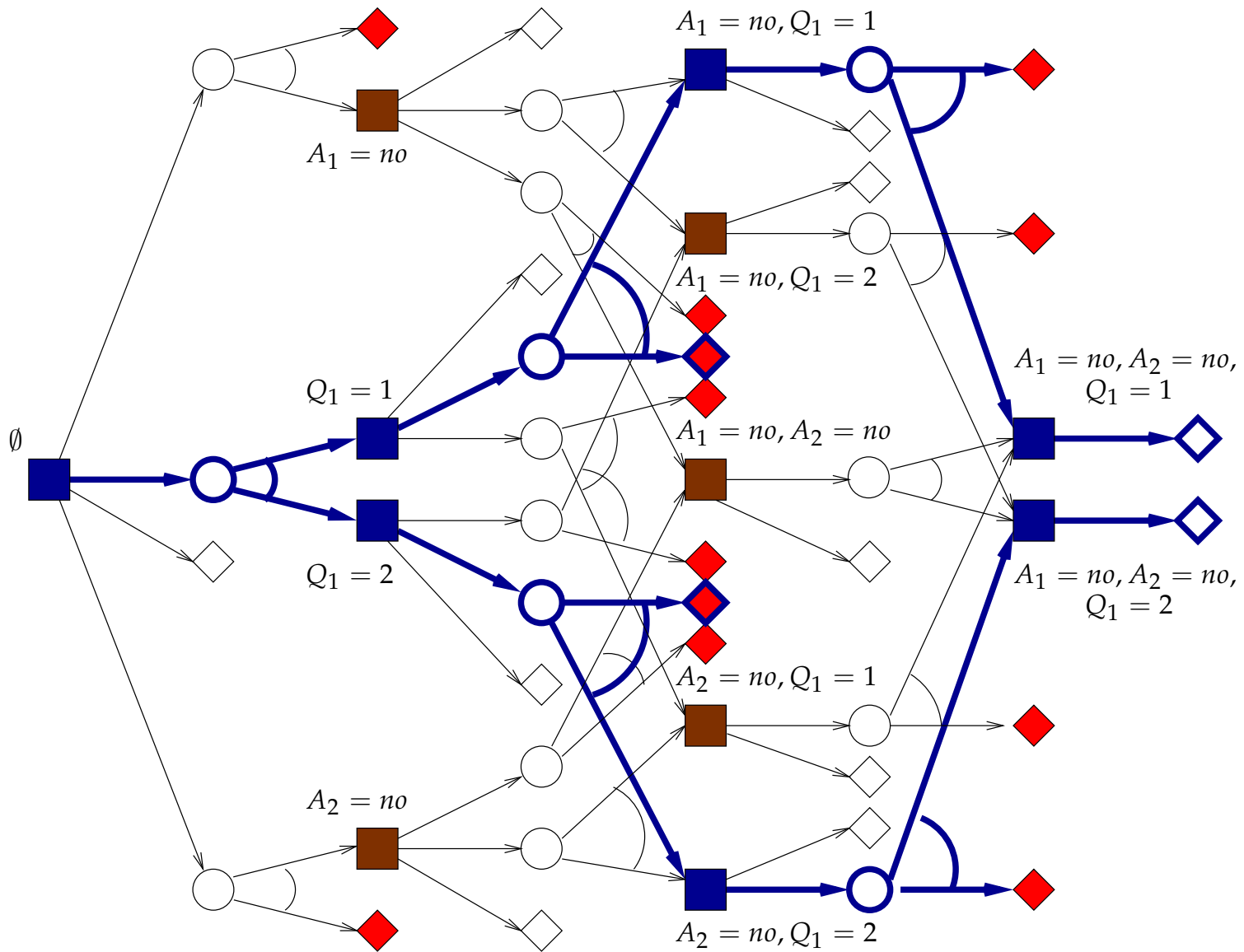
$$ECR(s^*) \leq ECR(s) .$$

Polynomiálně řešitelná úloha (S. Srinivas, 1995)

- (1) každá akce řeší právě jednu závadu,
- (2) všechny akce jsou navzájem nezávislé dano příčiny a
- (3) platí, že zařízení má v jednom okamžiku pouze jednu závadu (single fault assumption)

NP-těžká úloha (M. Vomlelová, 2000):

- (1) některé akce mohou řešit více než dvě závady



Algoritmy pro nalezení optimálního řešení

Metody prohledávání stavového prostoru všech řešení
(reprezentovaného AND/OR grafem):

- metoda větví a mezí (branch and bound)
- metody dynamického programování
- A^* pro AND/OR grafy popsány v knize:
J. Pearl, Heuristics: intelligent search strategies for computer
problem solving, 1984

Výpočetně náročné.

Suboptimální řešení v reálném čase

BATS troubleshooter:

- Vyvinutý v Laboratory for Normative Systems, v rámci společného projektu Hewlett-Packard a Aalborg University.
- Využívá několik heuristik založených na poměru p/c .
- Algoritmus je popsán v článku:
F. V. Jensen, U. Kjaerulff, B. Kristiansen, H. Langseth, C. Skaanning, J. Vomlel & M. Vomlelová: The SACSO methodology for troubleshooting complex systems. Special Issue on AI in Equipment Service, Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AIEDAM), Vol. 15, pp. 321-333, 2001.

Porovnání optimálního řešení s BATS troubleshooterem

Problem	\mathcal{A}	\mathcal{Q}	OPTIM	BATS	LBE	P/C
53	6	2	433.238	443.305	501.625	444.544
Tray	9	3	129.214	129.214	131.585	155.096
Overrun	11	3	106.204	112.456	117.377	116.801
Load	12	3	38.3777	38.4229	42.6062	43.0535
Pjam	13	4	124.323	124.365	299.415	300.855
Scatter	14	4	115.410	115.862	324.38	236.578
NotDupl	9	9	70.6740	73.5984	77.3768	121.098
Spots	16	5	161.385	162.246	863.362	286.749
MIO1	10	10	250.452	253.310	355.943	479.956