Experiments in runtime monitoring via probabilistic session types

@GSSI @UoM @DTU

C. Bartolo Burlò, A. Francalanza, A. Scalas,

C. Trubiani, E. Tuosto

IT-Matters 12/4/2021

To be presented @ COORDINATION 2021

What I'm going to talk about

- Binary session types in a nutshell
- A probabilistic variant
- Algorithmic derivation of (passive) monitors
    - are oblivious of participants' actual implementation
    - approximate participants' probabilistic observable behaviour
    - emit revocable judgements based on confidence intervals
- Ideas for future work

Covering my shoulders

"It is unanimously agreed that statistics depends somehow on probability. But, as to what probability is and how it is connected with statistics, there has seldom been such complete disagreement and breakdown of communication since the Tower of Babel."

Savage: Foundations of Statistics. Wiley, 1954

Binary session types in a nutshell

S runs a "lottery" game

C has to guess a number $1 \leq n \leq 100$ secretly chosen by S

seek for help

quit the game

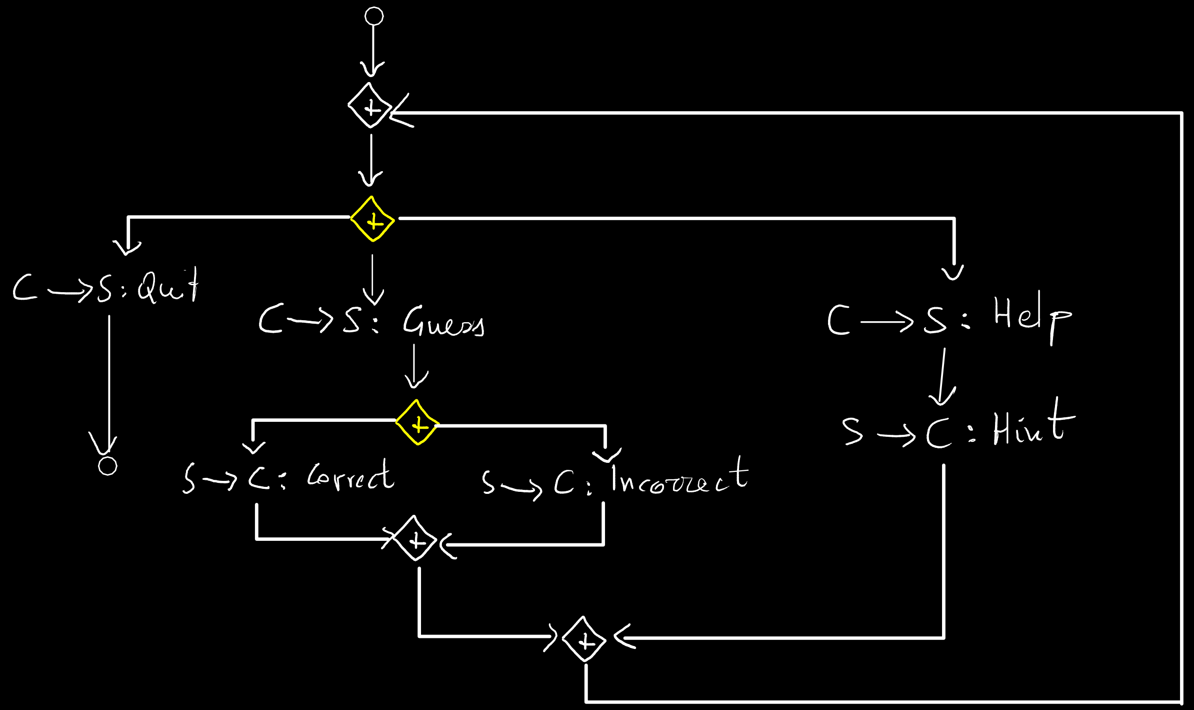
Binary session types in a nutshell

S runs a "Lottery" game

C has to guess a number $1 \leq n \leq 100$ secretly chosen by S

seek for help

quit the game



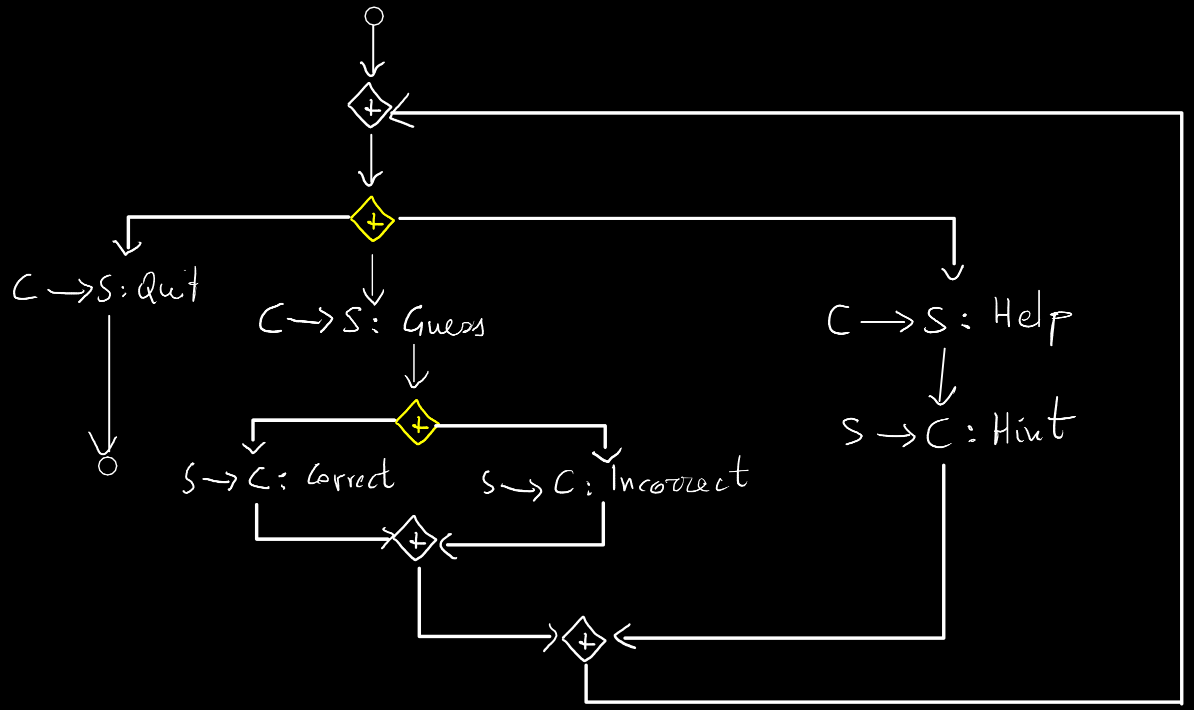
Binary session types in a nutshell

S runs a "Lottery" game

C has to guess a number $1 \leq n \leq 100$ secretly chosen by S

seek for help

quit the game



$$S = \underline{\text{rec}}\ X.\ \&\{\ ?\text{Guess}(\text{Int}).\ \oplus\{\ !\text{Correct}.\ X,\ !\text{Incorrect}.\ X\ \},\ ?\text{Help}.\ !\text{Hint}(\text{str}).\ X,\ ?\text{Quit}.\ \underline{\text{end}}\ \}$$

$$C = \underline{\text{rec}}\ X.\ \oplus\{\ !\text{Guess}(\text{Int}).\ \&\{\ ?\text{Correct}.\ X,\ ?\text{Incorrect}.\ X\ \},\ !\text{Help}.\ ?\text{Hint}(\text{str}).\ X,\ !\text{Quit}.\ \underline{\text{end}}\ \}$$

$$C = \overline{S}$$

Binary session types in a nutshell

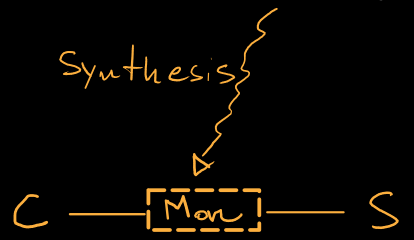
S runs a "Lottery" game

C has to guess a number $1 \leq n \leq 100$ secretly chosen by S

seek for help

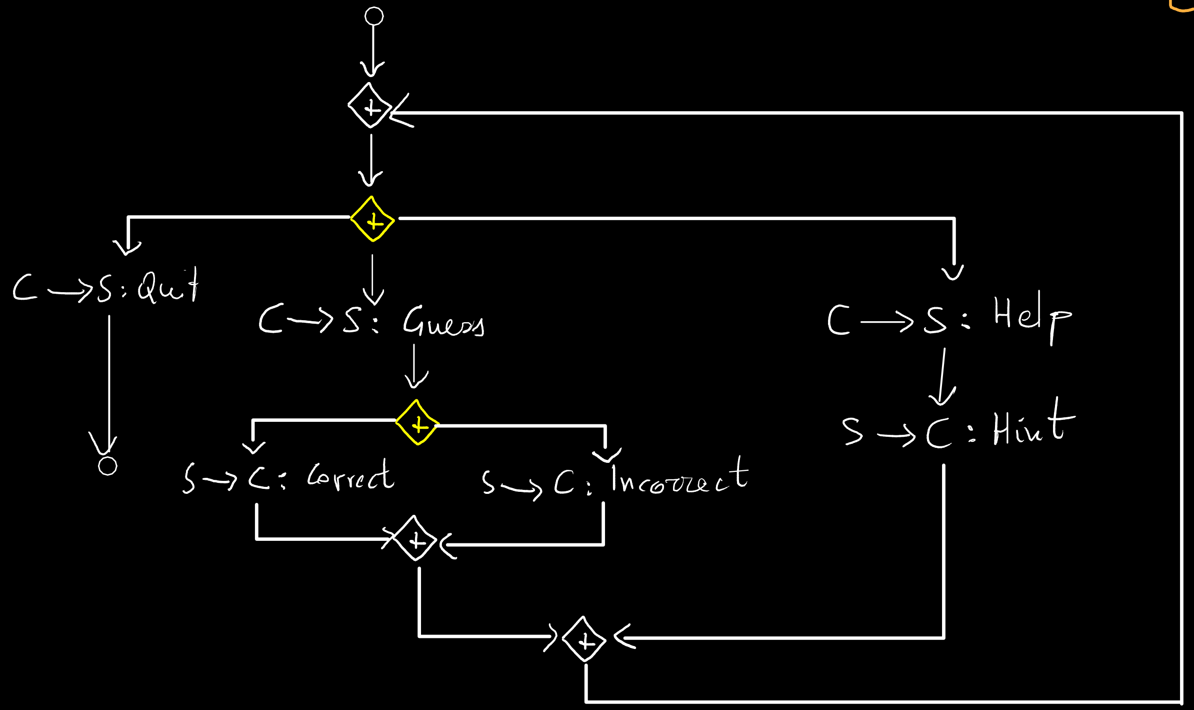
quit the game

$$S = \underline{rec}\ X.\ \&\{\ ?Guess(Int).\ \oplus\{\ !Correct.\ X,\ !Incorrect.\ X\ \},\ ?Help.\ !Hint(str).\ X,\ ?Quit.\ \underline{end}\ \}$$



$$C = \underline{rec}\ X.\ \oplus\{\ !Guess(Int).\ \&\{\ ?Correct.\ X,\ ?Incorrect.\ X\ \},\ !Help.\ ?Hint(str).\ X,\ !Quit.\ \underline{end}\ \}$$

$$C = \overline{S}$$



A probabilistic variant

$$S = \underline{\text{rec}}\ X.\ \&\{?\text{Guess}(\text{Int})[.75].\ \oplus\{!\text{Correct}[.01].\ X,\ !\text{Incorrect}[.99].\ X\},\ ?\text{Help}[.2].\ !\text{Hint}(\text{str}).\ X,\ ?\text{Quit}[.05].\ \underline{\text{end}}\}$$

prob. 1

$$T ::= \&\{?\ell_i(s_i)\ [p_i].T_i\}_{i\in I}\ \mid\ \oplus\{!\ell_i(s_i)\ [p_i].T_i\}_{i\in I}\ \mid\ \underline{\text{rec}}\ X.T\ \mid\ X\ \mid\ \underline{\text{end}}$$

$\forall i \in I: 0 < p_i \leq 1$

$\sum_{i\in I} p_i = 1$

A probabilistic variant

$$S = \underline{\text{rec}}\ X.\ \&\{ ?\text{Guess}(\text{Int})[.75].\ \oplus\{ !\text{Correct}[.01].\ X,\ !\text{Incorrect}[.99].\ X \},$$
$$?\text{Help}[.2].\ !\text{Hint}(\text{str}).\ X,$$
$$?\text{Quit}[.05].\ \underline{\text{end}} \}$$

prob. 1

$$T ::= \&\{ ?\ell_i(s_i)\,[p_i].T_i \}_{i\in I}$$
$$\mid\ \oplus\{ !\ell_i(s_i)\,[p_i].T_i \}_{i\in I}$$
$$\mid\ \underline{\text{rec}}\ X.T$$
$$\mid\ X$$
$$\mid\ \underline{\text{end}}$$

$\forall i \in I : 0 < p_i \leq 1$

$\sum_{i\in I} p_i = 1$

What is S actually specifying?

A probabilistic variant

$$S = \underline{\text{rec}}\ X.\ \&\{?\mathsf{Guess}(\mathsf{Int})[.75].\ \oplus\{!\mathsf{Correct}[.01].\ X,\ !\mathsf{Incorrect}[.99].\ X\},\ ?\mathsf{Help}[.2].\ !\mathsf{Hint}(\mathsf{str}).\ X,\ ?\mathsf{Quit}[.05].\ \underline{\text{end}}\}$$

$$\begin{aligned} T ::=\ & \&\{?\ell_i(s_i)[p_i].T_i\}_{i\in I} \\ |\ & \oplus\{!\ell_i(s_i)[p_i].T_i\}_{i\in I} \\ |\ & \underline{\text{rec}}\ X.T \\ |\ & X \\ |\ & \underline{\text{end}} \end{aligned}$$

What is S actually specifying?

```erlang
smartArse() ->
    receive
        {guess, G, C} -> C ! incorrect,
                         smartArse();
        help -> C ! "I wish I could help",
                smartArse();
        quit -> io:format("another sucker got it!")
    end;
```

Is this a good implementation?

Well, it type-checks, but...

"On-line" monitoring of probability

Let's consider a simple frequentist approach to estimate probabilities

```
S = rec X. & { ?Guess(Int)[.75].
                 ⊕ { !Correct[.01]. X,
                     !Incorrect[.99]. X
                   },
               ?Help[.2].!Hint(str). X,
               ?Quit[.05]. end
             }
```

$$
\begin{aligned}
S = \underline{\mathrm{rec}}\ X.\ \&\{ &?\mathrm{Guess}(\mathrm{Int})[.75].\ \oplus\{\ !\mathrm{Correct}[.01].\ X,\\
&\qquad\qquad\qquad\qquad\quad\ \ !\mathrm{Incorrect}[.99].\ X\\
&\qquad\qquad\qquad\qquad\},\\
&?\mathrm{Help}[.2].\ !\mathrm{Hint}(\mathrm{str}).\ X,\\
&?\mathrm{Quit}[.05].\ \underline{\mathrm{end}}\\
\}
\end{aligned}
$$

"On-line" monitoring of probability

Let's consider a simple frequentist approach to estimate probabilities

Our monitors

- have a parameter: confidence level $0 \le \ell \le 100$

$\mathcal{N}(0, 1)$

ℓ	Z-value
50	.645
68.269	1
90	1.6449
95	1.9599
99.99	4.4172

"On-line" monitoring of probability

$$
\begin{aligned}
S = \underline{\mathrm{rec}}\ X.\ \&\{ & ?\mathrm{Guess}(\mathrm{Int})[.75].\ \oplus\{ !\mathrm{Correct}[.01].\ X,\ !\mathrm{Incorrect}[.99].\ X \}, \\
& ?\mathrm{Help}[.2].\ !\mathrm{Hint}(\mathrm{str}).\ X, \\
& ?\mathrm{Quit}[.05].\ \underline{\mathrm{end}} \}
\end{aligned}
$$

Let's consider a simple frequentist approach to estimate probabilities

Our monitors

- have a parameter: confidence level $0 \leq \ell \leq 100$

$\mathcal{N}(0, 1)$

ℓ	Z-value
50	.645
68.269	1
90	1.6449
95	1.9599
99.99	4.4172

- Keep estimating probabilities:

$$\hat{p}_{ij} = \frac{\#\ \text{branch}_{ij}}{\#\ \text{choice}\ i}$$

eg

$$\hat{p}_{\&,\text{guess}} = \frac{\#\text{times C selected 'Guess'}}{\#\text{times C had to select}}$$

$$\hat{p}_{\oplus,\text{Incorrect}} = \frac{\#\text{times S replied incorrect}}{\#\text{times C guessed}}$$

"On-line" monitoring of probability

```
S = rec X. & { ?Guess(Int)[.75].
                 ⊕ { !Correct[.01]. X,
                     !Incorrect[.99]. X
                   },
               ?Help[.2].!Hint(str). X,
               ?Quit[.05].end
             }
```

Let's consider a simple frequentist approach to estimate probabilities

Our monitors

- have a parameter: confidence level $0 \le \ell \le 100$

$\mathcal{N}(0,1)$

ℓ	Z-value
50	.645
68.269	1
90	1.6449
95	1.9599
99.99	4.4172

- Keep estimating probabilities:

$$\hat{p}_{ij} = \frac{\#\ \text{branch}\ ij}{\#\ \text{choice}\ i}$$

eg

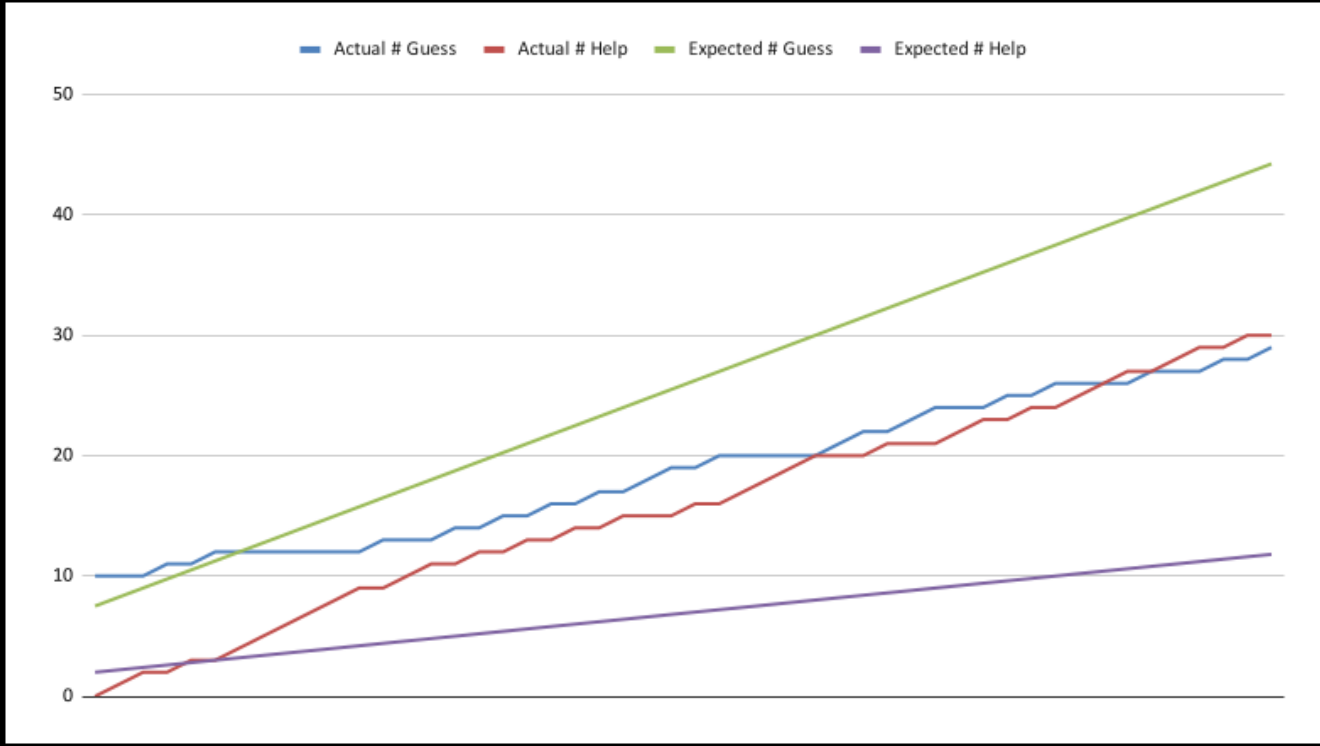
$$\hat{p}_{\&,\text{guess}} = \frac{\#\text{times C selected 'Guess'}}{\#\text{times C had to select}}$$

$$\hat{p}_{\oplus,\text{Incorrect}} = \frac{\#\text{times S replied incorrect}}{\#\text{times C guessed}}$$

- use $E_{ij} = Z(\ell)\sqrt{\dfrac{p_{ij}(1-p_{ij})}{\#\ \text{choice}\ i}}$ where p_{ij} is the specified prob. of the j-th branch of choice i

to flag warnings when $\hat{p}_{ij} \notin [p_{ij} - E_{ij},\ p_{ij} - E_{ij}]$ capping may be needed

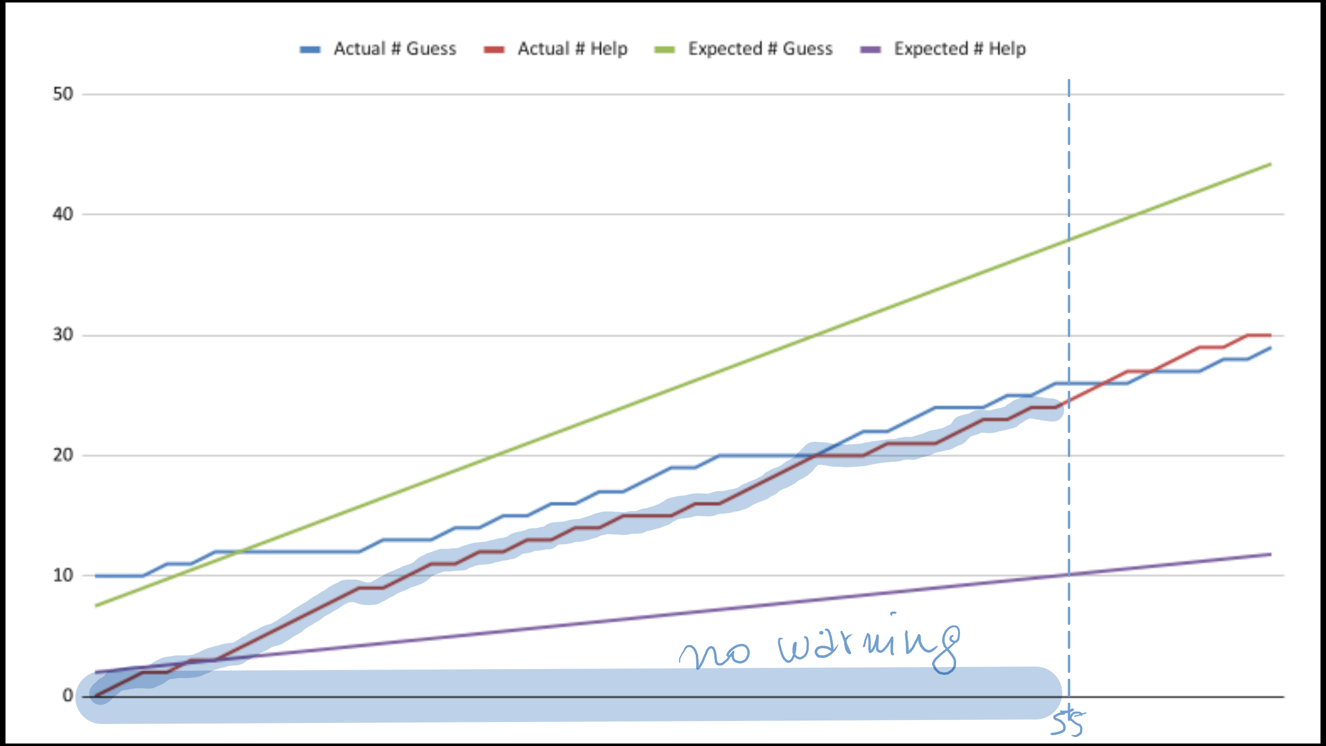
Example



$S = \text{rec } X. \,\&\{ ?\text{Guess}(\text{Int})[.75].\ \oplus\{ !\text{Correct}[.01].\,X,\ !\text{Incorrect}[.99].\,X \},$
$\quad ?\text{Help}[.2].\,!\text{Hint}(\text{str}).\,X,$
$\quad ?\text{Quit}[.05].\,\text{end} \}$

# iter.	# Guess	# Help	# Quit	p Guess	p Help	p Quit	Errors			Exp Guess	Exp Help
51	29	22	0	0.568627451	0.431372549	0	0.2678320907	0.2443004543	0.1369178325	38.25	10.2
52	30	22	0	0.576923076	0.423076923	0	0.2652442804	0.2419400081	0.1355949239	39	10.4
53	30	23	0	0.566037735	0.433962264	0	0.2627300598	0.2396466861	0.134309635	39.75	10.6
54	31	23	0	0.574074074	0.425925925	0	0.2602860062	0.2374173662	0.1330602158	40.5	10.8
55	31	24	0	0.563636363	0.436363636	0	0.2579089152	0.2352491256	0.1318450286	41.25	11
56	31	25	0	0.553571428	0.446428571	0	0.255595784	0.2331392254	0.1306625381	42	11.2
57	32	25	0	0.561403508	0.438596491	0	0.2533437949	0.2310850952	0.1295113039	42.75	11.4
58	33	25	0	0.568965517	0.431034482	0	0.2511503009	0.2290843208	0.1283899728	43.5	11.6
59	34	25	0	0.576271186	0.423728813	0	0.2490128126	0.2271346316	0.1272972723	44.25	11.8

Example



S = rec X. & {?Guess(Int)[.75].
⊕{ !Correct[.01]. X,
!Incorrect[.99]. X
},
?Help[.2].!Hint(str). X,
?Quit[.05].end
}

# iter.	# Guess	# Help	# Quit	$\hat{p}$ Guess	$\hat{p}$ Help	$\hat{p}$ Quit	Errors			Exp Guess	Exp Help
51	29	22	0	0.568627451	0.431372549	0	0.2678320907	0.2443004543	0.1369178325	38.25	10.2
52	30	22	0	0.576923076	0.423076923	0	0.2652442804	0.2419400081	0.1355949239	39	10.4
53	30	23	0	0.566037735	0.433962264	0	0.2627300598	0.2396466861	0.134309635	39.75	10.6
54	31	23	0	0.574074074	0.425925925	0	0.2602860062	0.2374173662	0.1330602158	40.5	10.8
55	31	24	0	0.563636363	0.436363636	0	0.2579089152	0.2352491256	0.1318450286	41.25	11
56	31	25	0	0.553571428	0.446428571	0	0.255595784	0.2331392254	0.1306625381	42	11.2
57	32	25	0	0.561403508	0.438596491	0	0.2533437949	0.2310850952	0.1295113039	42.75	11.4
58	33	25	0	0.568965517	0.431034482	0	0.2511503009	0.2290843208	0.1283899728	43.5	11.6
59	34	25	0	0.576271186	0.423728813	0	0.2490128126	0.2271346316	0.1272972723	44.25	11.8

Example

$$
\begin{aligned}
S = \underline{\text{rec}}\, X.\ \&\{ ?\text{Guess}(\text{Int})[.75].\ & \oplus\{\ !\text{Correct}[.01].\ X, \\
& \phantom{\oplus\{}\ !\text{Incorrect}[.99].\ X \\
& \}, \\
?\text{Help}[.2].\ !\text{Hint}(\text{str}).\ X, & \\
?\text{Quit}[.05].\ \underline{\text{end}} & \\
\} &
\end{aligned}
$$

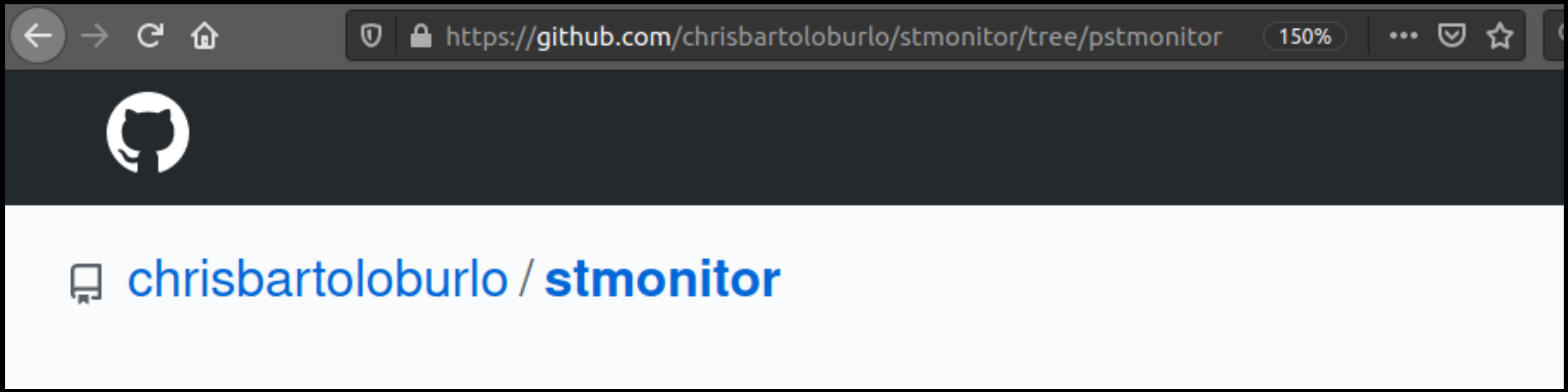
# iter.	# Guess	# Help	# Quit	$\hat{p}$ Guess	$\hat{p}$ Help	$\hat{p}$ Quit	Errors			Exp Guess	Exp Help
51	29	22	0	0.568627451	0.431372549	0	0.2678320907	0.2443004543	0.1369178325	38.25	10.2
52	30	22	0	0.576923076	0.423076923	0	0.2652442804	0.2419400081	0.1355949239	39	10.4
53	30	23	0	0.566037735	0.433962264	0	0.2627300598	0.2396466861	0.134309635	39.75	10.6
54	31	23	0	0.574074074	0.425925925	0	0.2602860062	0.2374173662	0.1330602158	40.5	10.8
55	31	24	0	0.563636363	0.436363636	0	0.2579089152	0.2352491256	0.1318450286	41.25	11
56	31	25	0	0.553571428	0.446428571	0	0.255595784	0.2331392254	0.1306625381	42	11.2
57	32	25	0	0.561403508	0.438596491	0	0.2533437949	0.2310850952	0.1295113039	42.75	11.4
58	33	25	0	0.568965517	0.431034482	0	0.2511503009	0.2290843208	0.1283899728	43.5	11.6
59	34	25	0	0.576271186	0.423728813	0	0.2490128126	0.2271346316	0.1272972723	44.25	11.8

Wrapping up

- Applications (?)
  - Support to take decisions (eg flagging possible fraud detections)
  - Checking AI's learning
  - Flagging faulty components

- Future work
  - Try Different estimators & different warning policies
  - Non-uniform confidence levels
  - "Global" probabilistic behaviour (Prob ("a big purchase shortly after small ones") is very small)
  - Can we formalise monitor's correctness?
  - What is monitorable?
  - When are constraints on probabilities inconsistent?

Thanks!


https://github.com/chrisbartoloburlo/stmonitor/tree/pstmonitor
150%
chrisbartoloburlo / stmonitor