UiO : University of Oslo

Declarative Design of Spoken Dialogue Systems with Probabilistic Rules

Pierre Lison
Language Technology Group,
University of Oslo

September 20, 2012
SemDial

Introduction

- Spoken dialogue systems typically rely on pipeline architectures with «black-box» components developed separately
- Each component employs ad-hoc encoding formats for their inputs/outputs and internal parameters
- Formats rarely compatible with one another!
    - Difficult to derive a semantic interpretation as a whole
    - Difficult to perform *joint* optimisations
    - Domain- or task-specific knowledge often «hardwired»

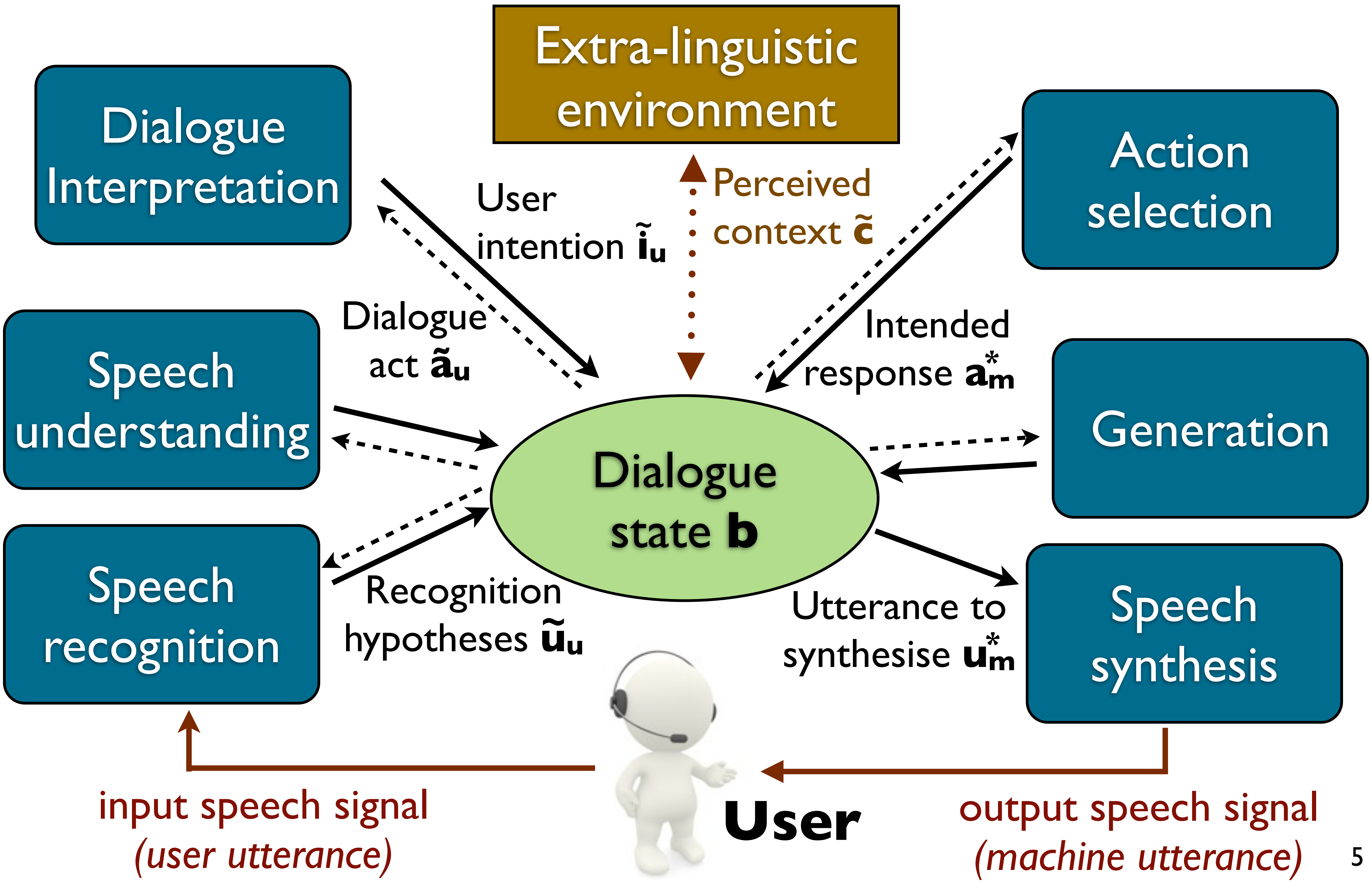
Introduction

- We adopt an alternative approach:
  - Declarative specification of all domain- & task-specific knowledge via a *common representation formalism*
  - System architecture «stripped down» to a core set of algorithms for probabilistic inference
- Advantages:
  - Domain portability
  - More transparent semantics
  - More flexible workflow

General architecture

- *Blackboard architecture* revolving around a shared dialogue state
  - Dialogue models are attached to this dialogue state, and listen for relevant changes appearing on it
  - When triggered, they read/write to this state, creating and updating the state variables
- Dialogue state encoded as a *Bayesian Network*
  - Each network node represents a distinct state variable, possibly connected to other variables

General architecture



Dialogue models

- The dialogue models are all expressed in terms of probabilistic rules
- Probabilistic rules are *high-level templates* for constructing probabilistic models
- Why use this representation formalism?
  - Take advantage of the *internal structure* of the problem while retaining the stochastic modelling
  - Abstraction mechanism (reduced set of parameters)

Probabilistic rules

- Probabilistic rules take the form of structured **if...then...else** cases
- Mapping from *conditions* to (probabilistic) *effects*:

if (condition$_1$ holds) **then**
  P(effect$_1$)= θ_1, P(effect$_2$)= θ_2, ...
else if (condition$_2$ holds) **then**
  P(effect$_3$) = θ_3, ...
...
else
  P(effect$_n$) = θ_n, ...

Probabilistic rules

- *Conditions* are (arbitrarily complex) logical formulae on state variables
- *Effects* are value assignments on state variables
- Effect probabilities are *parameters* that can be estimated from data

Example:

if (a_m = AskRepeat) **then**
  $P(a_u' = a_u) = 0.9$
  $P(a_u' \neq a_u) = 0.1$

Utility rules

- The formalism can also describe *utility models*
- In this case, the rule maps each condition to an assignment of *utility values* for particular actions:

if (condition$_1$ holds) **then**
  $Q(\text{actions}_1) = \theta_1$, $Q(\text{actions}_2) = \theta_2$, ...
else if (condition$_2$ holds) **then**
  $Q(\text{actions}_3) = \theta_3$, ...
...
else
  $Q(\text{actions}_n) = \theta_n$, ...

Rule instantiation

- How are the rules applied to the dialogue state?
- The rules are *instantiated* in the Bayesian Network, expanding it with new nodes and dependencies

Rule instantiation

- How are the rules applied to the dialogue state?
- The rules are *instantiated* in the Bayesian Network, expanding it with new nodes and dependencies

r_1:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$P(V = \ldots \wedge W = \ldots) = 0.6$

(The ... dots in r_1 should be replaced by concrete values)

Rule instantiation

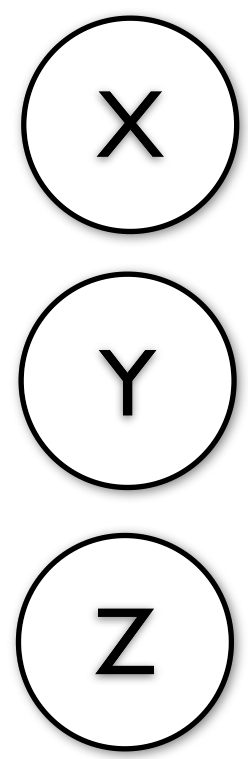
- How are the rules applied to the dialogue state?
- The rules are *instantiated* in the Bayesian Network, expanding it with new nodes and dependencies

r_1:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$P(V = \ldots \wedge W = \ldots) = 0.6$

(The ... dots in r_1 should be replaced by concrete values)



Rule instantiation

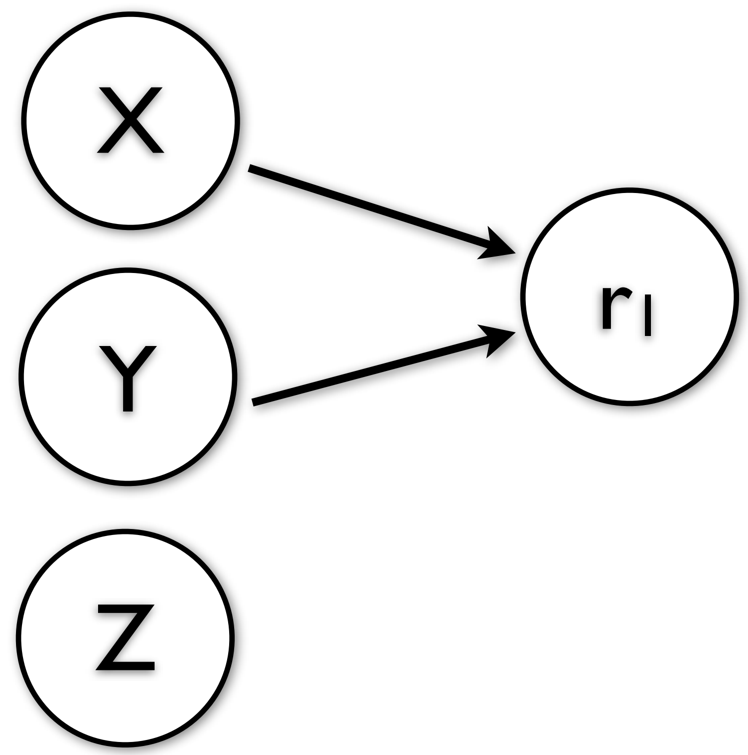
- How are the rules applied to the dialogue state?
- The rules are *instantiated* in the Bayesian Network, expanding it with new nodes and dependencies

r_1:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$P(V = \ldots \wedge W = \ldots) = 0.6$

(The ... dots in r_1 should be replaced by concrete values)



Rule instantiation

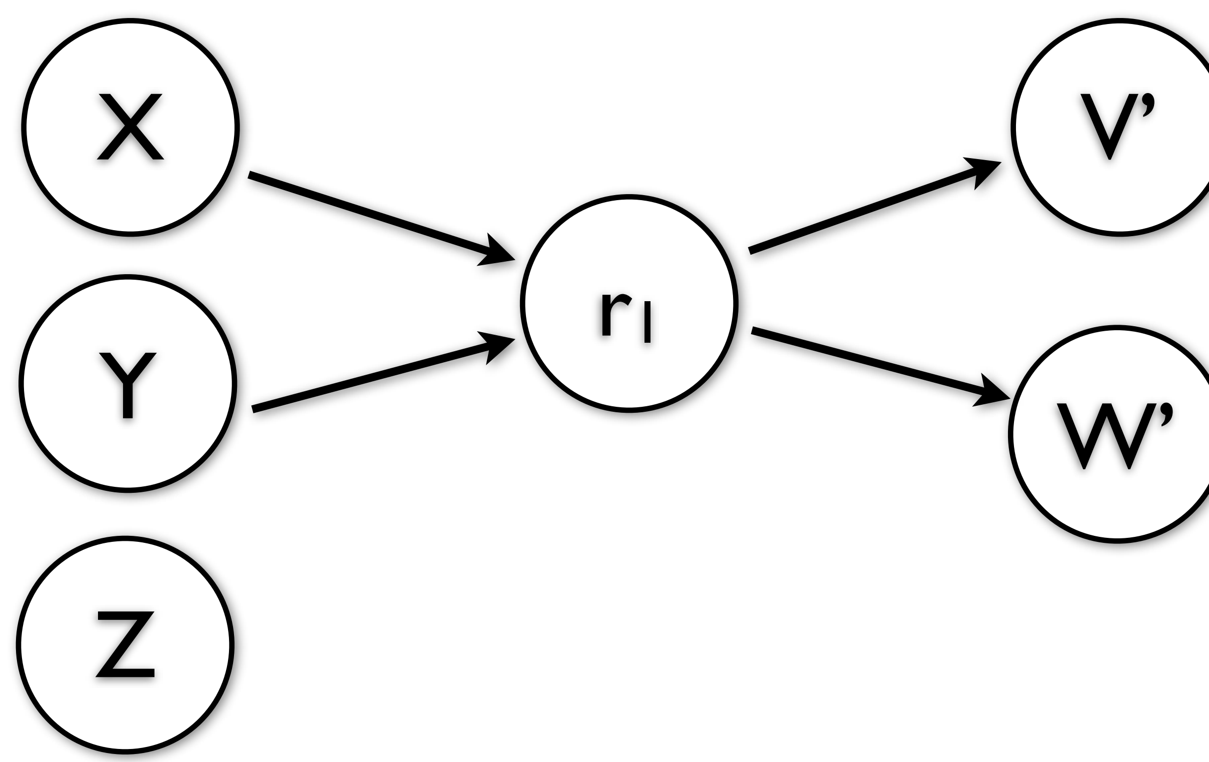
- How are the rules applied to the dialogue state?
- The rules are *instantiated* in the Bayesian Network, expanding it with new nodes and dependencies

r_1:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$P(V = \ldots \wedge W = \ldots) = 0.6$

(The ... dots in r_1 should be replaced by concrete values)



Rule instantiation

- The instantiation procedure is similar for utility rules, although one must employ utility and decision nodes:

Rule instantiation

- The instantiation procedure is similar for utility rules, although one must employ utility and decision nodes:

r_2:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$Q(A_1 = \ldots \wedge A_2 = \ldots) = 3$

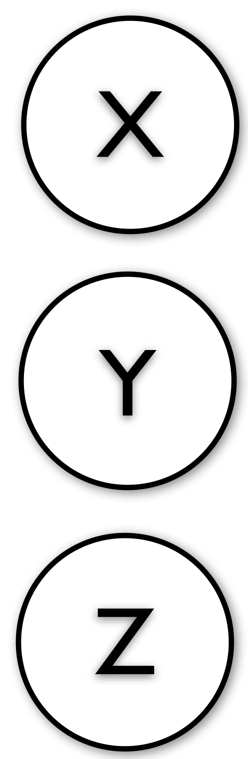
Rule instantiation

- The instantiation procedure is similar for utility rules, although one must employ utility and decision nodes:

r_2:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$Q(A_1 = \ldots \wedge A_2 = \ldots) = 3$



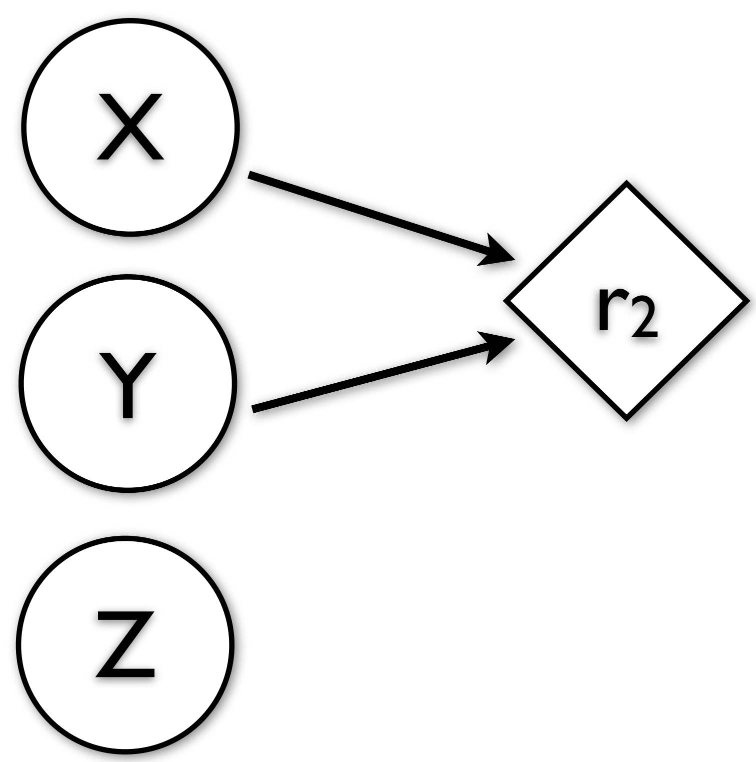
Rule instantiation

- The instantiation procedure is similar for utility rules, although one must employ utility and decision nodes:

r_2:

if $(X = \ldots \vee Y \neq \ldots)$ **then**

$Q(A_1 = \ldots \wedge A_2 = \ldots) = 3$

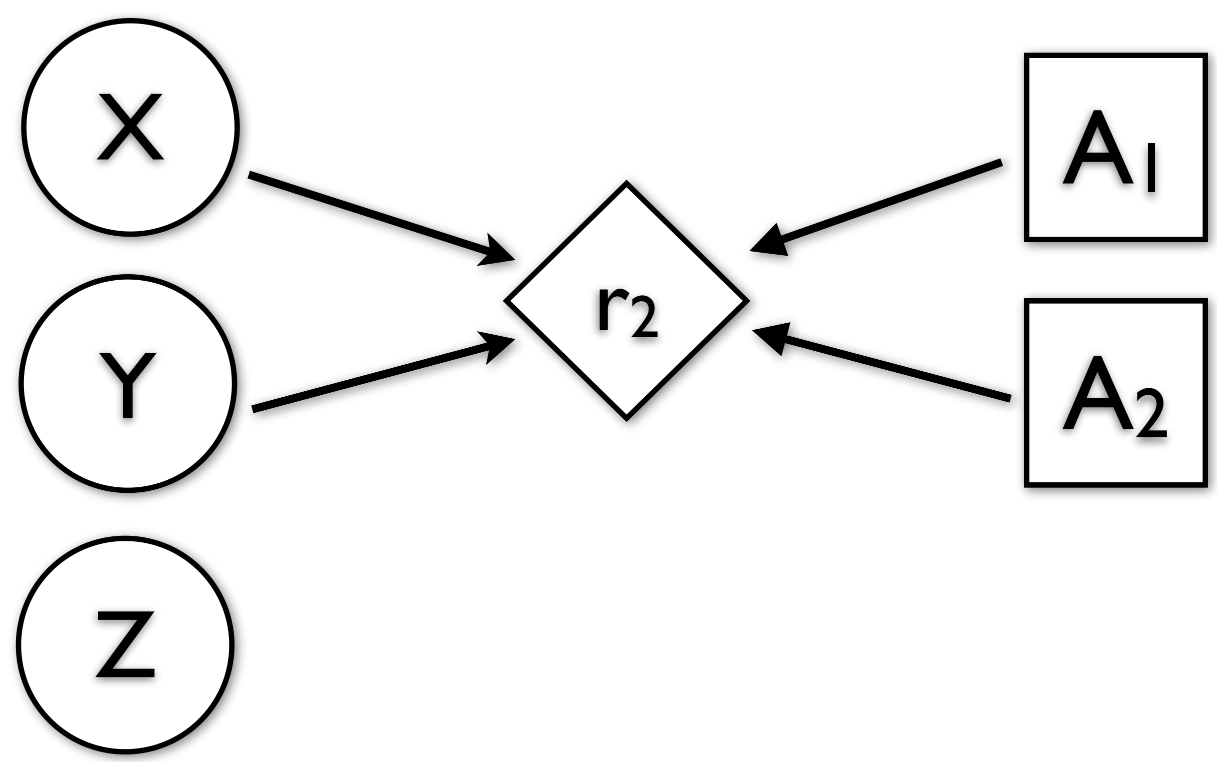


Rule instantiation

- The instantiation procedure is similar for utility rules, although one must employ utility and decision nodes:

r_2:

if ($X = \ldots \vee Y \neq \ldots$) **then**
$Q(A_1 = \ldots \wedge A_2 = \ldots) = 3$

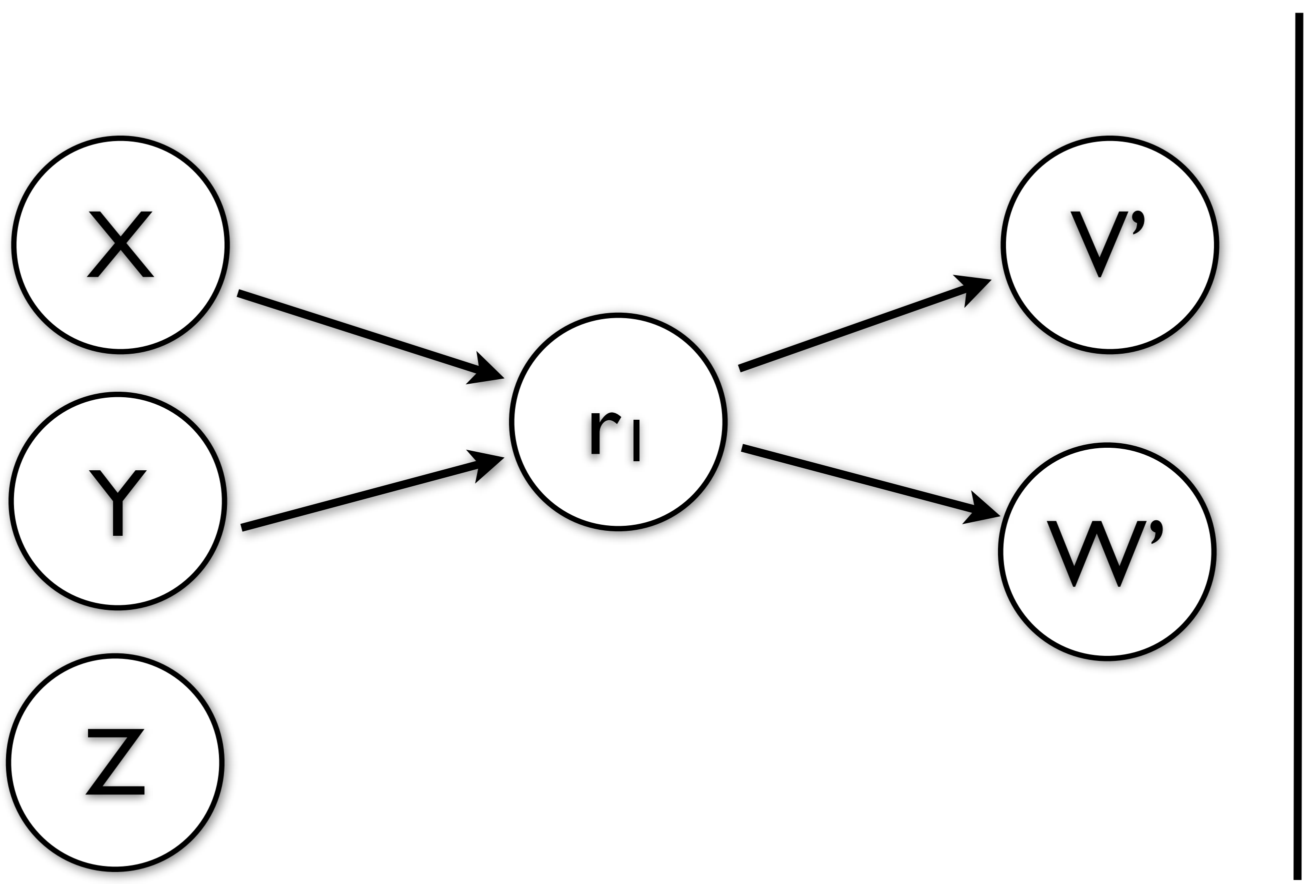


Rule instantiation

- If the rule parameters (probabilities or utilities) are uncertain, we add other nodes expressing their distribution

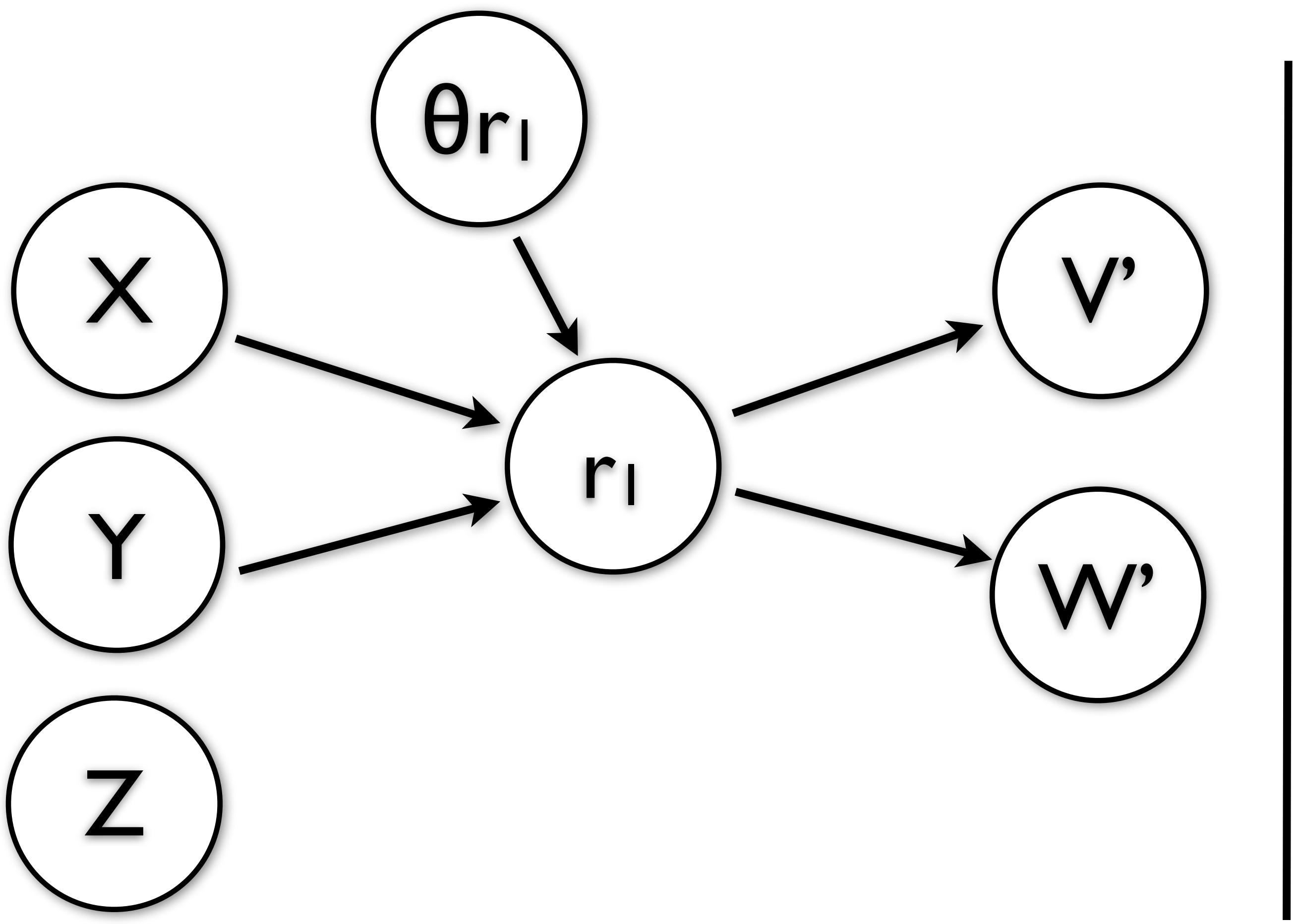
Rule instantiation

- If the rule parameters (probabilities or utilities) are uncertain, we add other nodes expressing their distribution



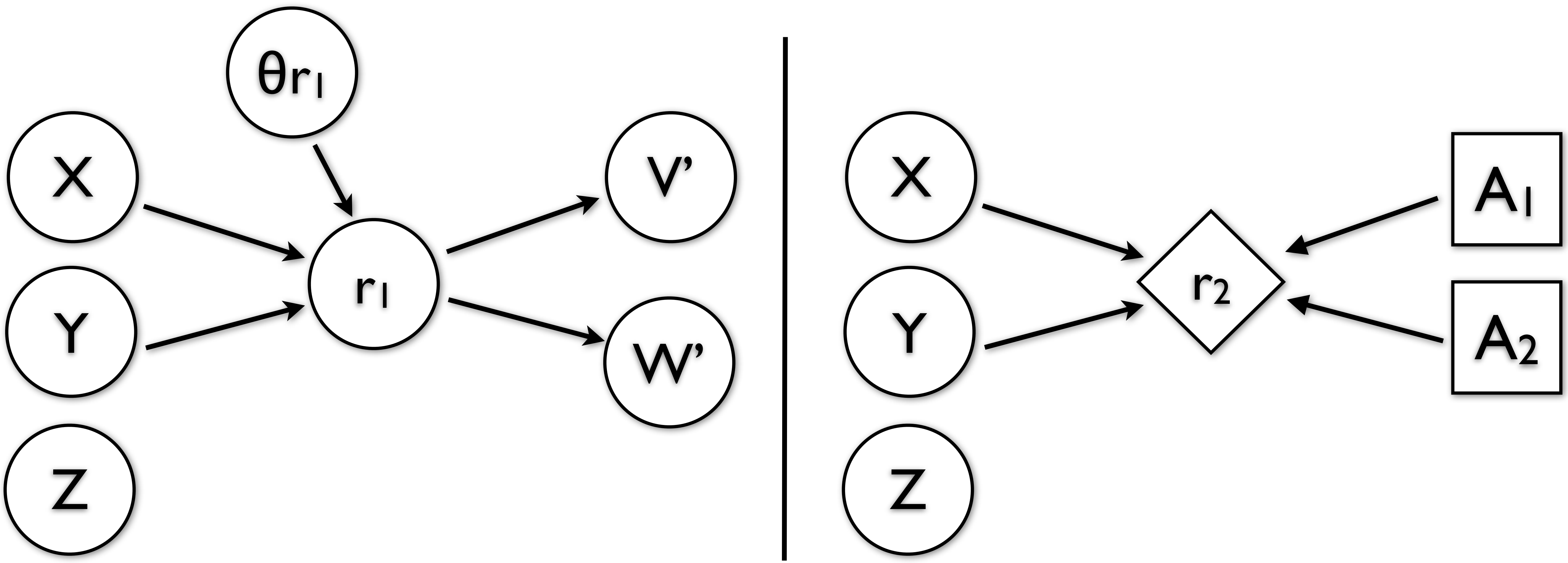
Rule instantiation

- If the rule parameters (probabilities or utilities) are uncertain, we add other nodes expressing their distribution



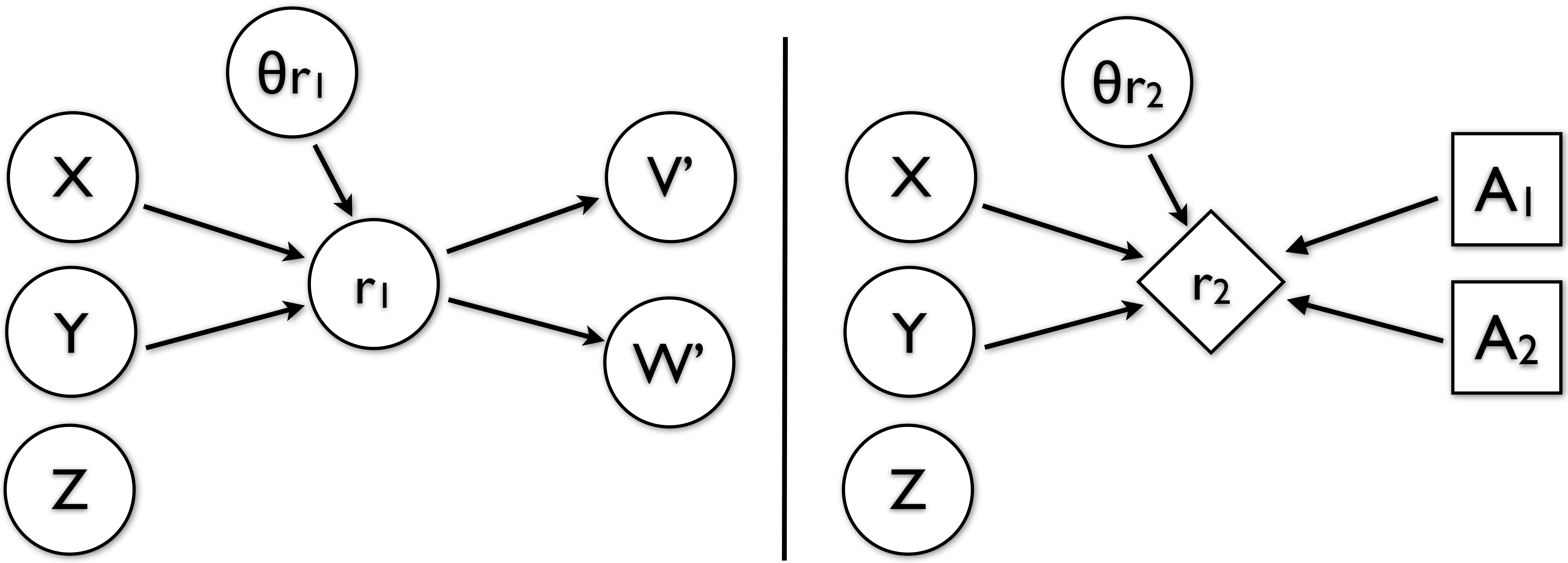
Rule instantiation

- If the rule parameters (probabilities or utilities) are uncertain, we add other nodes expressing their distribution



Rule instantiation

- If the rule parameters (probabilities or utilities) are uncertain, we add other nodes expressing their distribution

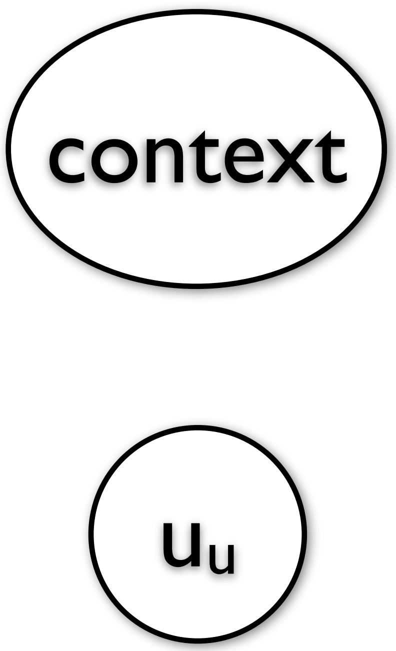


Processing workflow

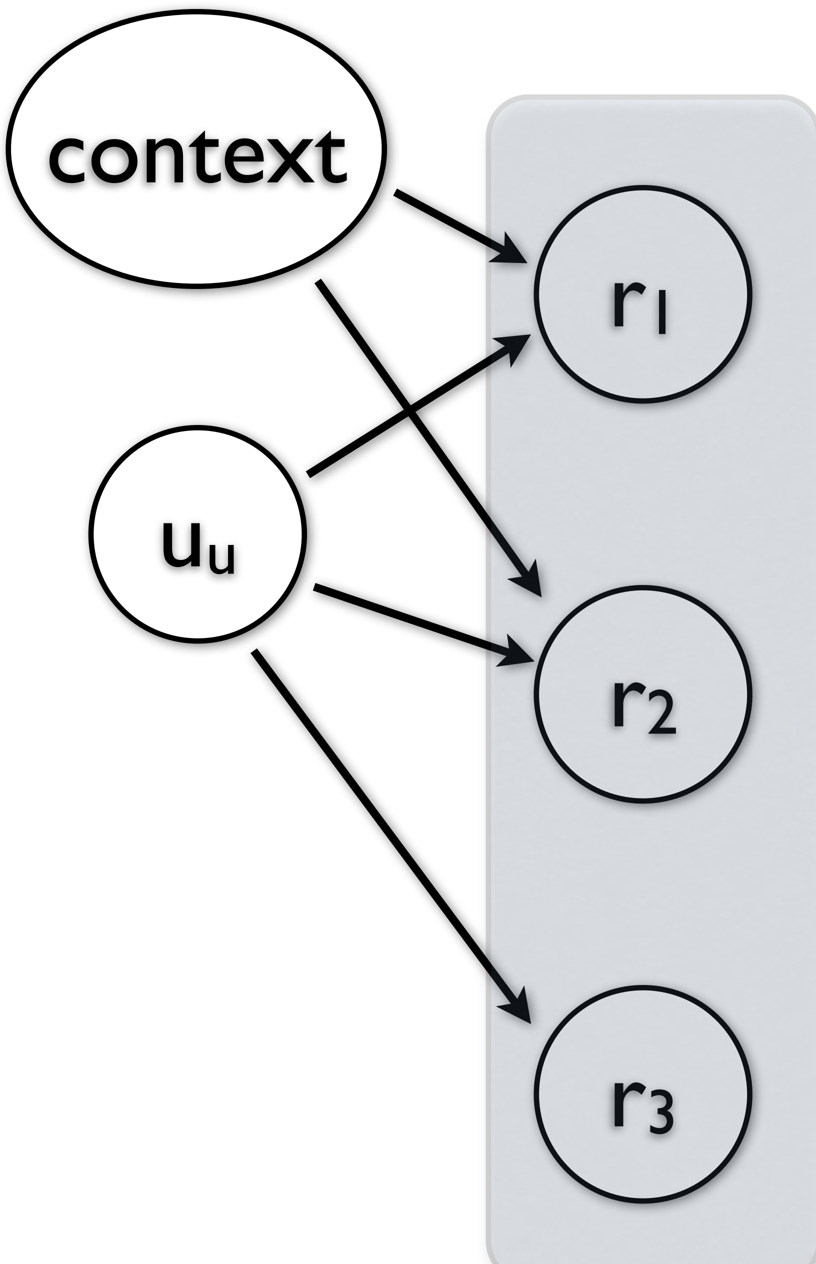
- To ease the domain design, the rules are grouped into *models*
- Each model is associated with a *trigger variable* causing its activation
- When a model is activated:
  - A rule node is created for each rule, conditionally dependent on the variables used in the conditions
  - Nodes corresponding to the output variables of the rule are also created/updated, and connected to the rule node

Processing workflow (example)

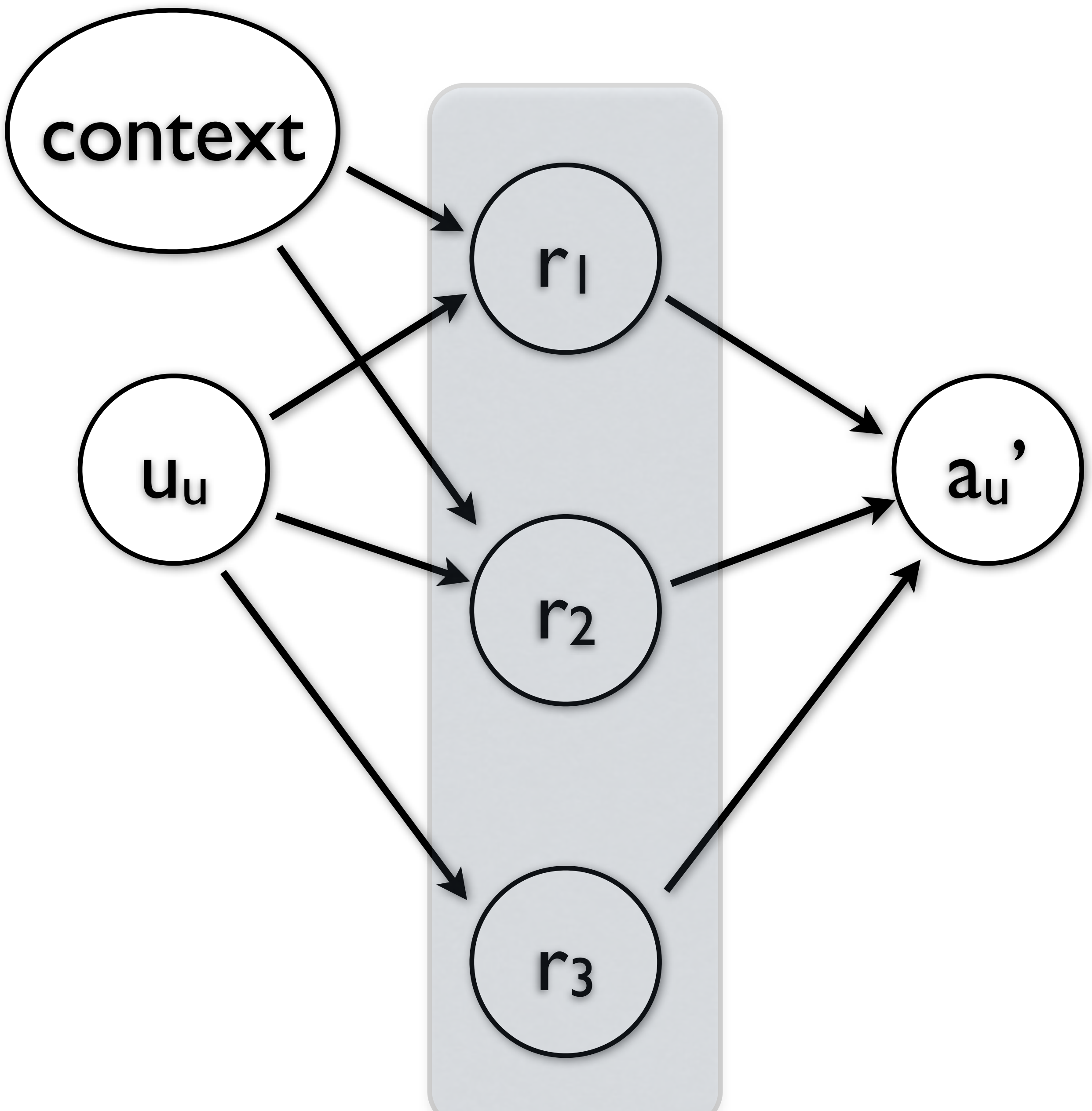
Processing workflow (example)



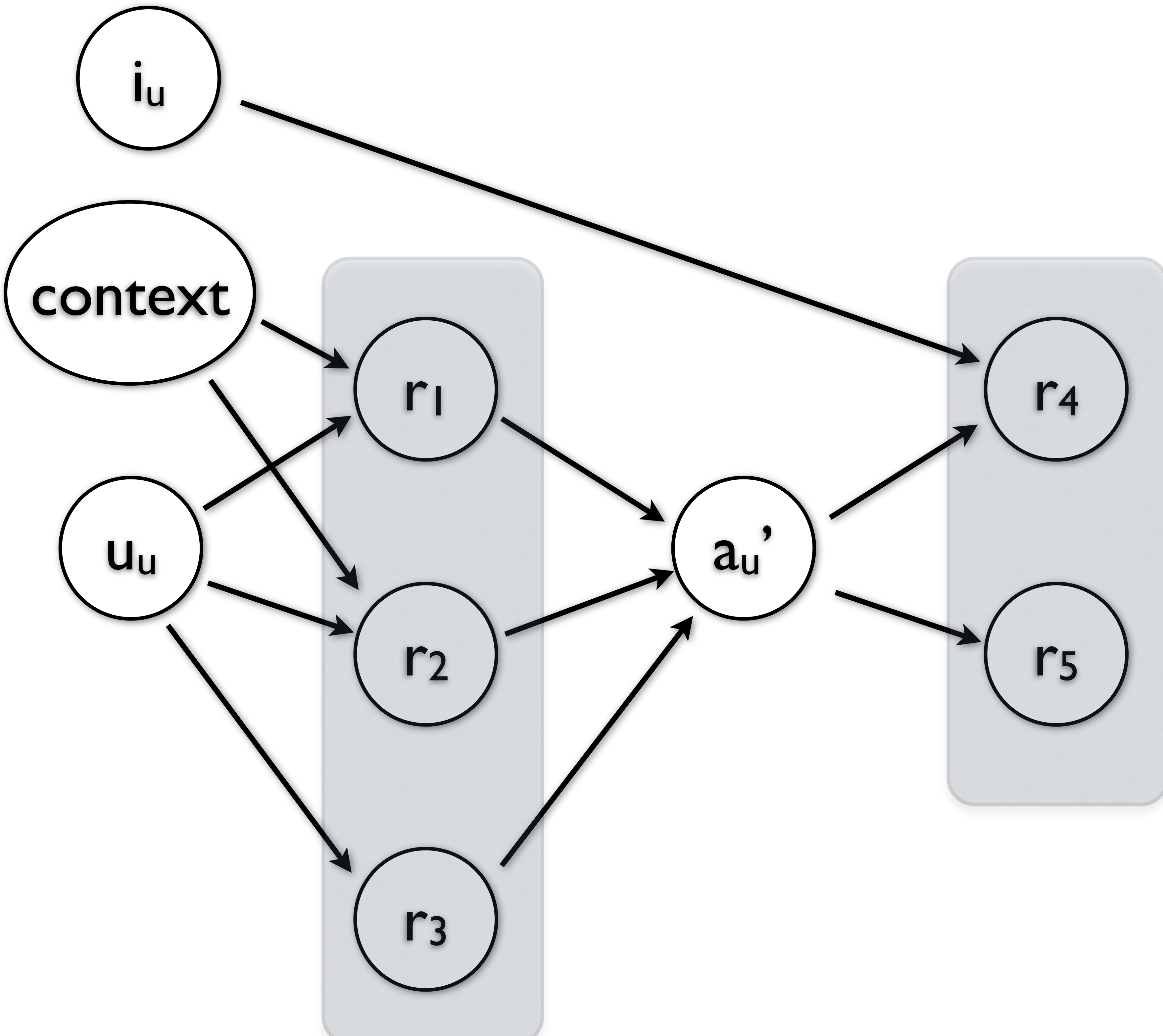
Processing workflow (example)



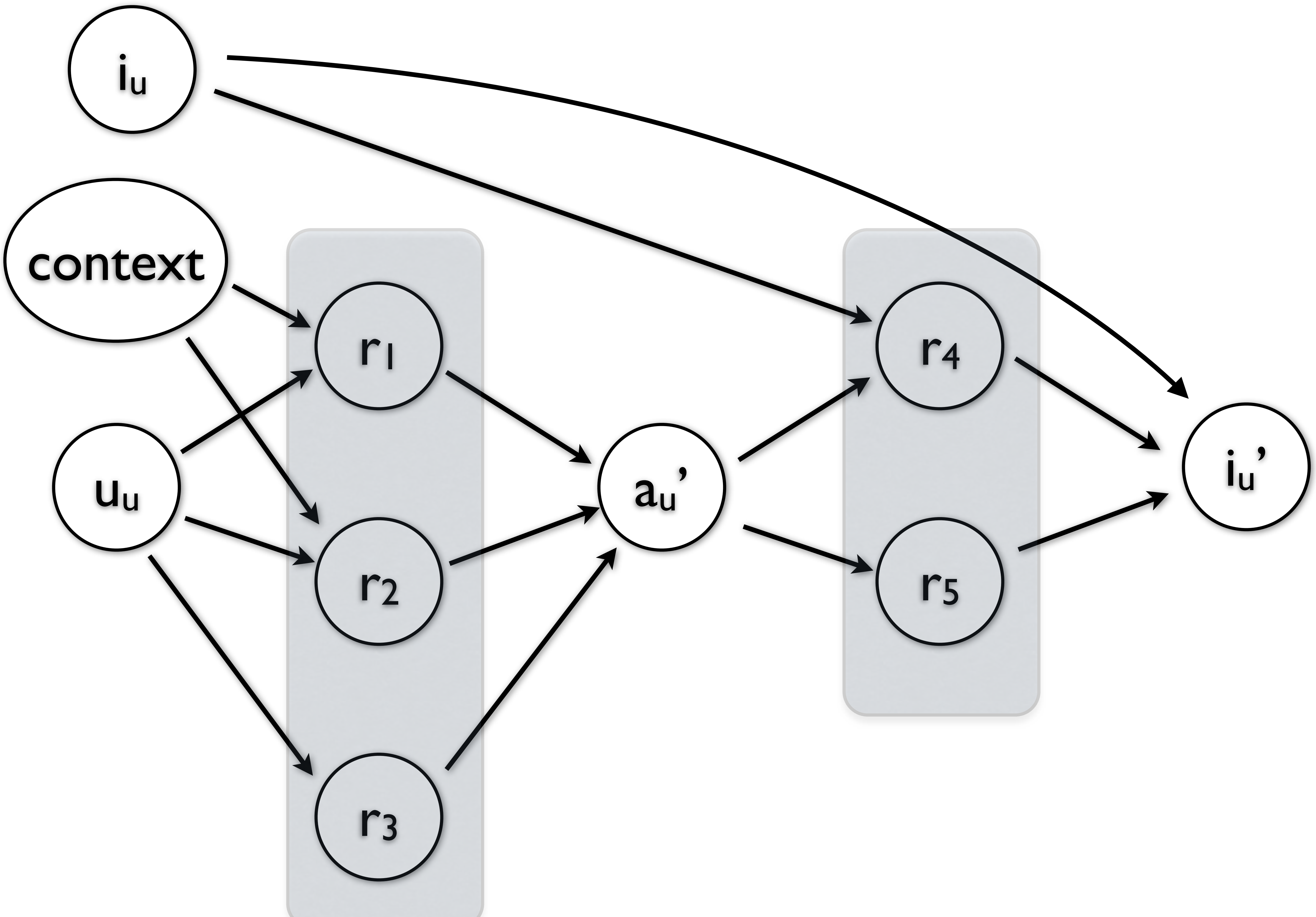
Processing workflow (example)



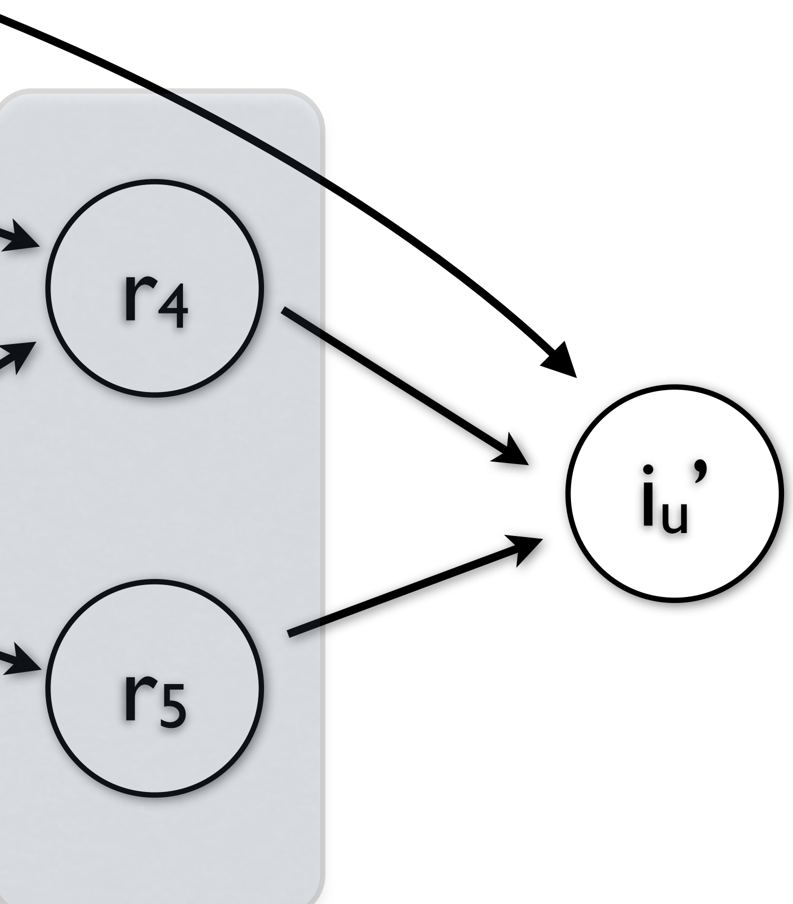
Processing workflow (example)



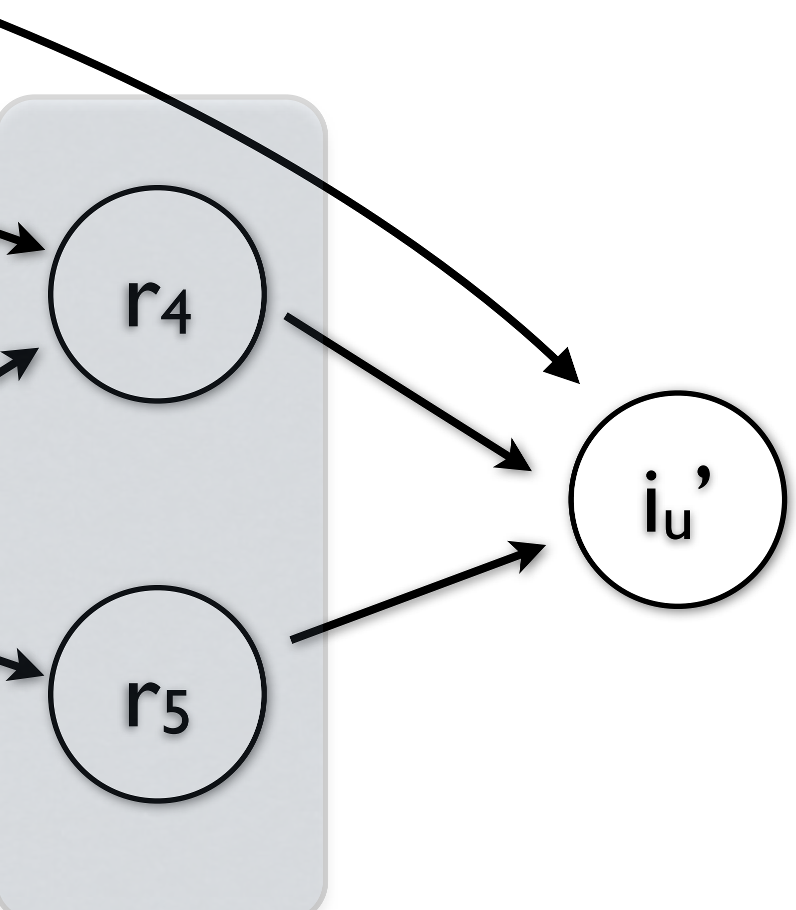
Processing workflow (example)



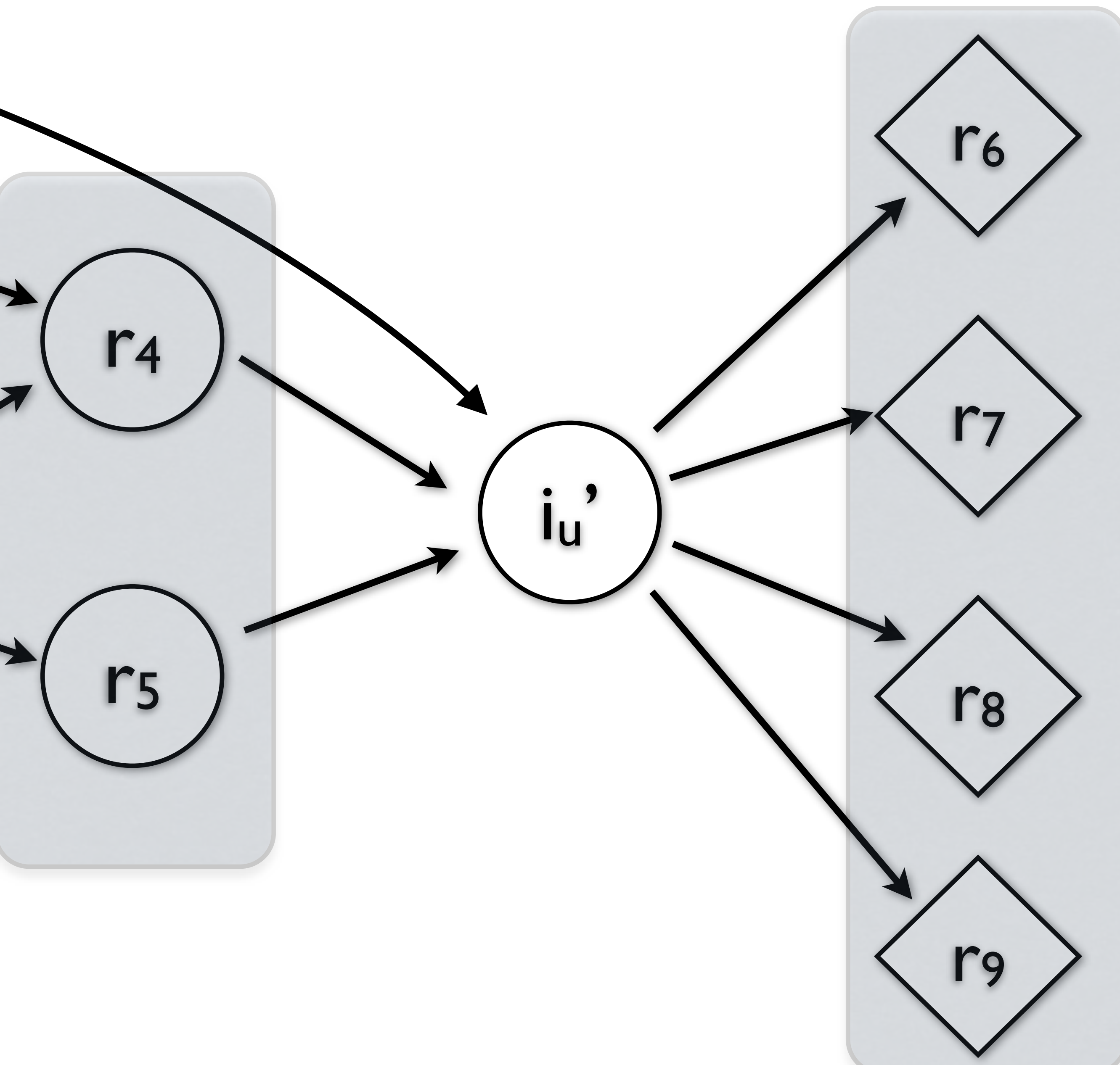
Processing workflow (example)



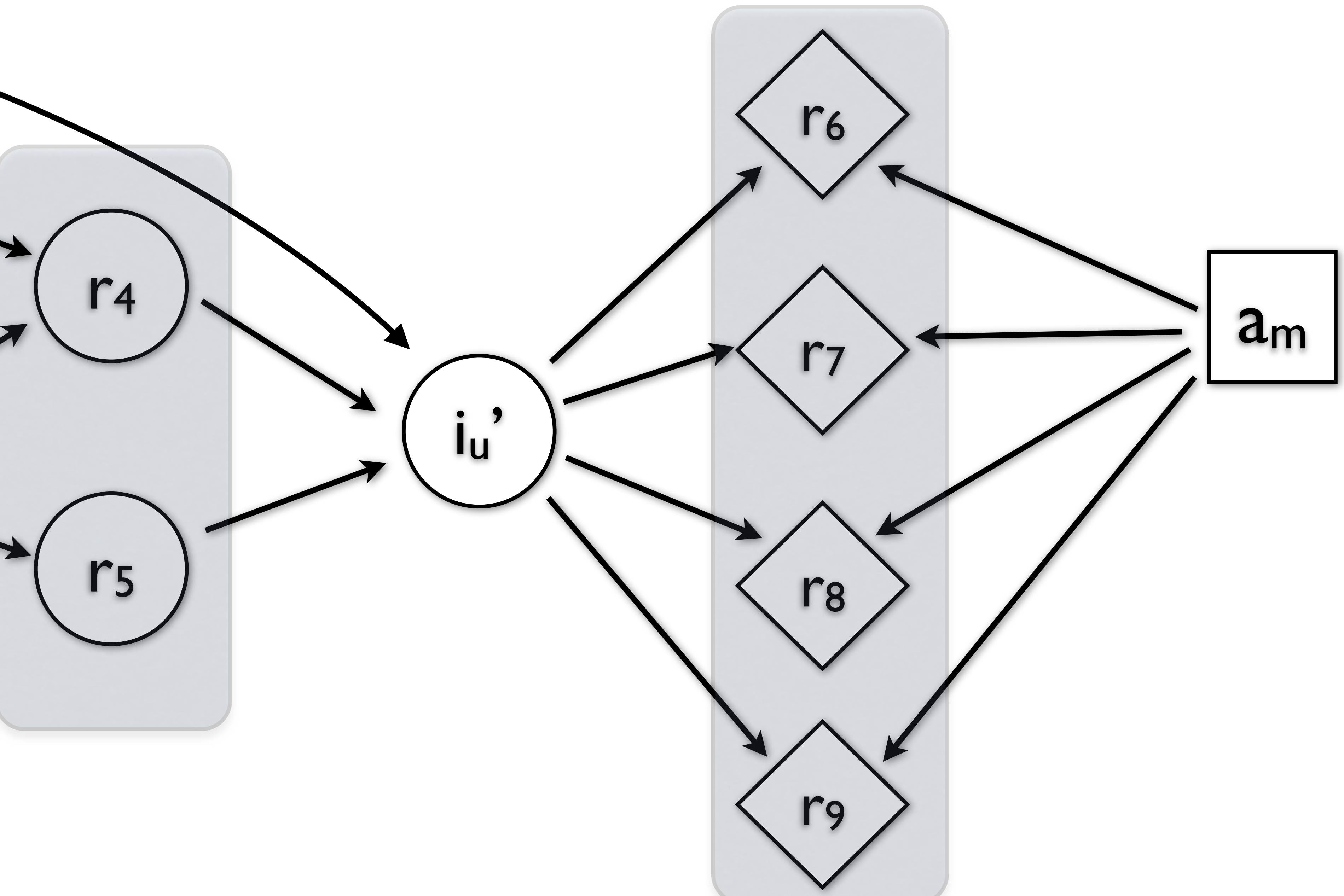
Processing workflow (example)



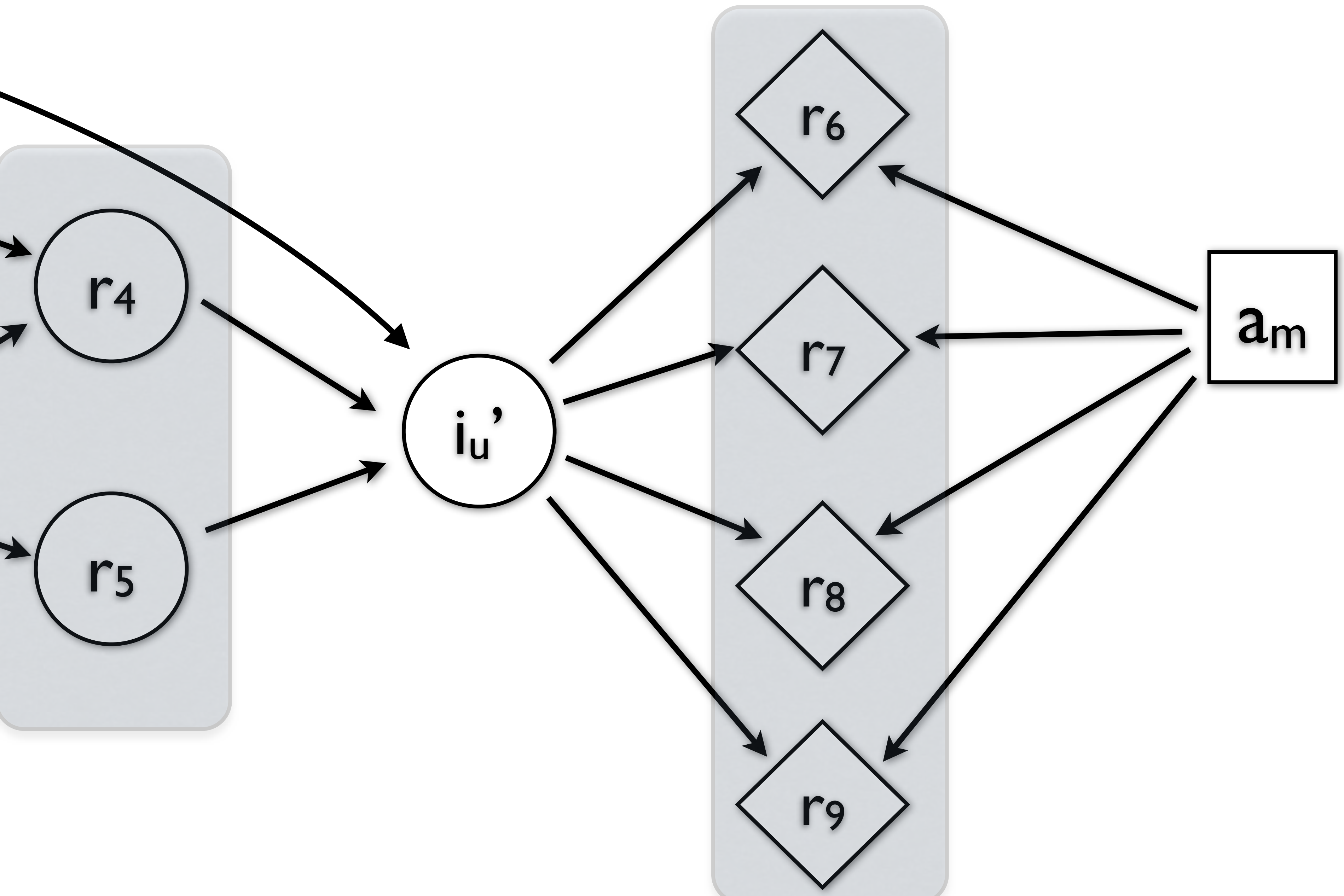
Processing workflow (example)



Processing workflow (example)



Processing workflow (example)



etc.

Processing workflow

- Additional details
  - No pipeline restriction: processing flow is possible
  - Decision nodes require a *decision* to be made, by selecting the value with maximum utility
  - Once the dialogue state is «stable» (no more model can be triggered), it is pruned to reduce it to a minimal size, retaining only the necessary nodes
  - The rules update existing variables or create *new* ones

Experiments

- The described formalism was implemented and tested in a simple human-robot interaction scenario
- The models for NLU, DM and NLG were encoded as probabilistic rules (total of 68 rules)

Experiments

- The utilities for the action selection rules were learned from Wizard-of-Oz data
- The other rules (NLU and NLG) were deterministic
- System also included a speech recogniser, TTS, and libraries for controlling the physical actions of the robot

[Pierre Lison, «Probabilistic Dialogue Models with Prior Domain Knowledge», SIGDIAL 2012]

Examples

- Dialogue act recognition rule:

$$
\begin{aligned}
r_1 :\ & \textbf{if } (u_u \text{ matches "left arm down"}) \\
& \vee (u_u \text{ matches "lower * left arm"}) \\
& \vee (u_u \text{ matches "down * left arm"}) \ \textbf{then} \\
& \quad \{P(a'_u = \text{LeftArmDown}) = 1.0\}
\end{aligned}
$$

- Prediction of next user action:

$$
\begin{aligned}
r_2 :\ & \textbf{if } (a_m = \text{AskRepeat}) \ \textbf{then} \\
& \quad \{P(a'_u = a_u) = 0.9\}
\end{aligned}
$$

Examples

- Action selection rules:

$$r_3: \textbf{if } (i_u = \text{RequestMovement}(\text{X})) \textbf{ then}$$
$$\{Q(a'_m = \text{DoMovement}(\text{X})) = 3.0\}$$

$$r_4: \textbf{if } (true) \textbf{ then}$$
$$\{Q(a'_m = \text{AskRepeat}) = 1.2\}$$

- Natural language generation rule:

$$r_5: \textbf{if } (a_m = \text{Ack}) \textbf{ then}$$
$$\{Q(u'_m = \text{“ok”}) = 1.0 \ \wedge$$
$$Q(u'_m = \text{“great”}) = 1.0 \ \wedge$$
$$Q(u'_m = \text{“thanks”}) = 1.0\}$$

Conclusions

- Dialogue system design based on the specification of probabilistic rules
- «Hybrid» approach combining domain knowledge and stochastic modelling
- Step towards a cleaner separation between system architecture and domain- and task-specific knowledge?

Future work

- Online estimation of the rule parameters (e.g. model-based Bayesian reinforcement learning)
- Joint optimisations of the parameters for NLU, DM and NLG models
- Incremental processing

Next interaction domain

Next interaction domain