



Knowledge Brokering in AKT

Stephen Potter

Centre for Intelligent Systems and their Applications
Division of Informatics, University of Edinburgh



Introduction



- AKT vision of problem-solving within a distributed environment of service providers.
- Basic need to match customer service requirements with the appropriate providers.
- Some sort of brokering agent is required:
 - providers advertise their services;
 - customers post their service requests;
 - broker has ability to match requests with services.

AKT Brokering Services



- Two complementary brokering algorithms, operating at different levels of abstraction:
- ‘Functional’ brokering:
 - Services described at low level of abstraction.
 - Returns precise results to precise queries.
 - Results in the form of alternative executable ‘plans’.
- ‘Transformational’ brokering:
 - Services described at a more general level.
 - Returns less specific plans in response to less specific queries.
- *No single ‘brokering algorithm’?*

'Functional' Brokering



- Services described through:
 - Agent-Name and URI;
 - Ontology;
 - List of competences:
 - Competence name;
 - Preconditions;
 - Input/output 'roles';
 - Required external competences.
- (Environment also contains ontology 'servers' and 'bridges'...)
- A query is in form of a request for a particular named competence.

Describing Services



```
service(  
  agent-name,  
  agent-uri,  
  agent-ontology,  
  [  
    competence((name ← preconditions),  
                input-roles,  
                output-roles,  
                external-competences),  
    competence(...),  
    ...  
  ]).
```

Agent *ticket-office*



```
service(  
  ticket-office,  
  http://oronsay.aiai.ed.ac.uk:12100,  
  travel-ontology,  
  [  
    competence((issue_ticket(Person,italy,Ticket) ← has_money(Person,1000,gb_pound)),  
      [person(Person)],  
      [valid_ticket(Ticket)],  
      [pay(Person,ticket_office,1000,gb_pound)]),  
  
    competence((issue_ticket(Person,spain,Ticket) ← has_money(Person,1500,euro)),  
      [person(Person)],  
      [valid_ticket(Ticket)],  
      [pay(Person,ticket_office,1500,euro)]),  
  
    competence((issue_ticket(Person,egypt,Ticket) ← has_visa(Person,egypt),  
      has_money(Person,1200,egyptian_pound),  
      has_passport(Person)),  
      [person(Person)],  
      [valid_ticket(Ticket)],  
      [pay(Person,ticket_office,1200,egyptian_pound)]),  
  ]).
```

Plan Synthesis



- Plan synthesis is a recursive procedure.
- Starting with an empty plan, the broker:
 - Looks for an agent described as offering the desired competence;
 - Are all preconditions & external requirements met?
 - If so, stop, and add this agent to the current plan;
 - If not, search for other agents offering these as competences.
- With backtracking, can generate alternative plans.

Synthesised Plan



```
ask(traveller,has_money(joe,1000,gb_pound),
      http://foula.aiai.ed.ac.uk:9000),
test([person(joe)]),
ask(ticket_office,(issue_ticket(joe,italy,_Ticket):-
      has_money(joe,1000,gb_pound)),
      http://oronsay.aiai.ed.ac.uk:12100,
      using(traveller,pay(joe,ticket_office,1000,gb_pound),
      http://foula.aiai.ed.ac.uk:9000)),
test([valid_ticket(_Ticket)])
```


Transformational Brokering



- An attempt to operate at a more abstract service level.
- A service is described as a transformation from one or more input 'bodies of knowledge' to an output body of knowledge:
 - A transformation is described according to a given classification.
 - A body of knowledge is described in terms of its representation format and abstraction level.
- A query takes the form of the description of a (possibly partial, possibly more general) desired transformation.

Service Advertisement



Knowledge Transformation Agent Panel

File Help

Agent address:
http://oronsay.aiai.ed.ac.uk:9000

Query/Service name:
neural-network-training-service

Broker address:
http://oronsay.aiai.ed.ac.uk:8005

Service request
 Service advertisement

AKT

- Knowledge_transformation_type
 - Transformations_for_knowledge_structuring
 - Transformations_for_knowledge_sharing
 - Semantic_alteration
 - Segmentation
 - Generalisation_synthesis
 - View_generation
 - Abstraction
 - Semantic_reconciliation
 - Syntactic_reconciliation
 - Transformations_for_knowledge_reuse

Knowledge transformation type:
Generalisation_synthesis

Input knowledge:
examples Instance_abstraction_level Numeric-value_pairs

Output knowledge:
trained-network Exclusive_domain_abstraction_level Connectionist_network

status

Service Request



Knowledge Transformation Agent Panel

File Help

Agent address:
http://oronsay.aiai.ed.ac.uk:9000

Query/Service name:
gen-query

Broker address:
http://oronsay.aiai.ed.ac.uk:8005

Service request
 Service advertisement

AKT

- Knowledge_transformation_type
 - Transformations_for_knowledge_structuring
 - Transformations_for_knowledge_sharing
 - Semantic_alteration
 - Segmentation
 - Generalisation_synthesis
 - View_generation
 - Abstraction
 - Semantic_reconciliation
 - Syntactic_reconciliation
 - Transformations_for_knowledge_reuse

Knowledge transformation type:
Generalisation_synthesis

Input knowledge:
input-knowledge1 Instance_abstraction_level Logical_representation

Add input knowledge **Clear input knowledge**

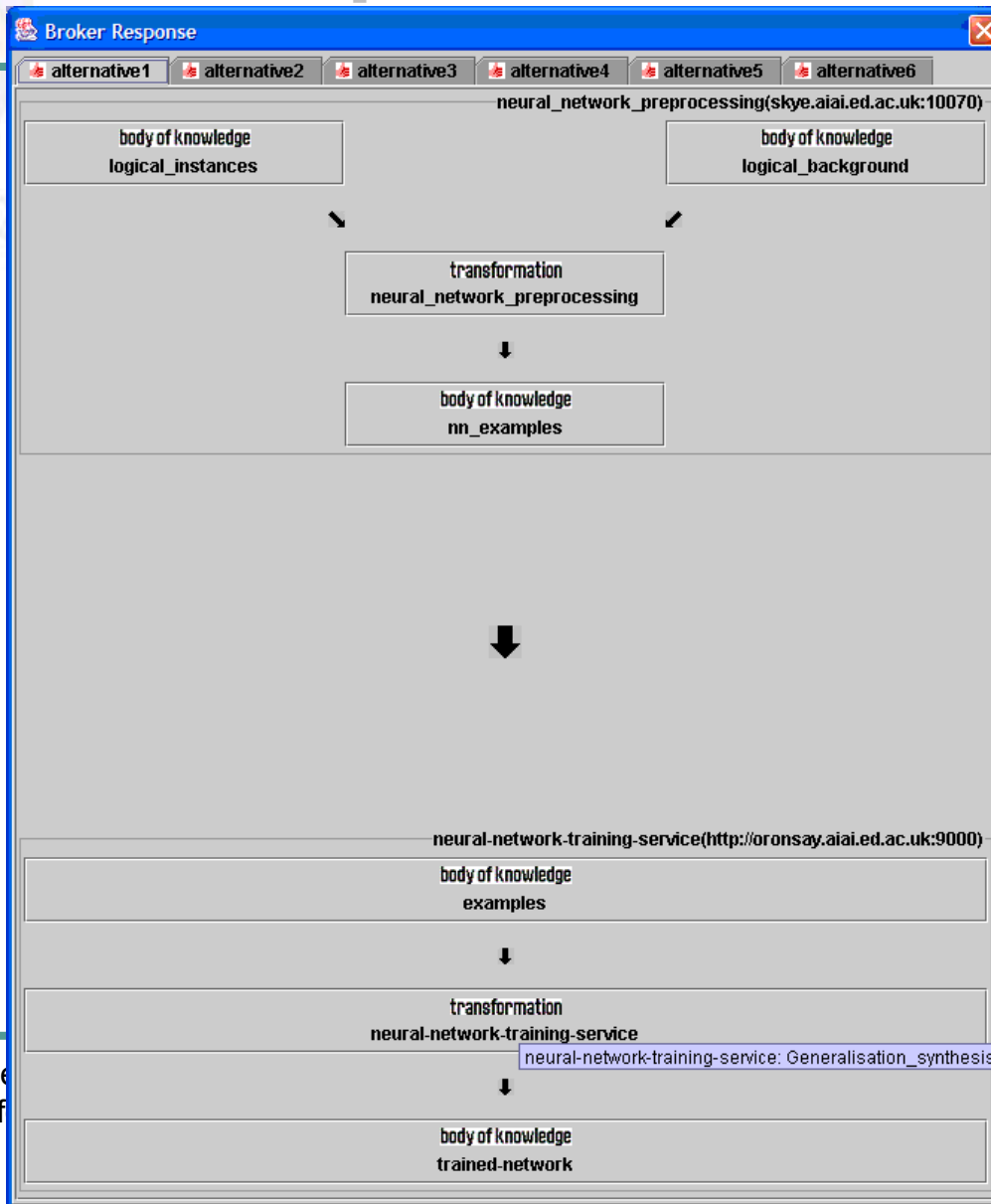
Output knowledge:

Add output knowledge **Clear output knowledge**

Send to broker **Cancel**

status
Generated RDF, sent to broker
Waiting for response

Sample Solution



Centre for Intelligent
Division of Information



Issues



- What sort of broker is appropriate?
 - Depends on the problem...and the assumptions that can be made about the environment and its agents....
- Broker mechanisms in Prolog:
 - Will this scale to more agents doing more complicated things?
- Interoperability:
 - Broker communications described using FIPA with content expressed in RDF(S).
 - How feasible is the use of shared ontologies for service/query content?
- Security? Trust? Reliability? etc...
 - Of services? Of broker?