Appendix 1: Kparser output on "hand grasp knife and it cut bowl"

Figure 1: Kparser parsing of the phrases: "hand grasp knife and it cut bowl"
Appendix 2: Kparser output on "The left hand was grasping the ruler."

Figure 2: K-parser parsing of the phrases: "The left hand was grasping the ruler."
Appendix 3 - ASP code of Section 4

time(1..3).
fluent(in(X,Y)) :- object(X), object(Y).
object(tofu,knife,bowl).
appears(put,tofu,bowl,1).
appears(cut,knife,bowl,2).

artifact(knife).
artifact(bowl).

occurs(A,S,O,T) :- appears(A,S,O,T),
not ab(A,S,O,T).

ab(cut,S,O,T) :- time(T), artifact(S), artifact(O).

occurs(cut,S,O,T) :- time(T), appears(cut,S,OO,T),
ab(cut,S,OO,T), holds(in(O,OO),T).

holds(in(X,Y),T+1) :- time(T), occurs(put,X,Y,T).

holds(F,T+1) :- fluent(F), time(T), holds(F,T), not nholds(F,T+1).

#hide time(X).
#hide fluent(X).
#hide object(X).
#hide appears(X,Y,Z,T).
#hide artifact(X).
#hide ab(X,Y,Z,T).

% Answer set contains
% occurs(put,tofu,bowl,1) holds(in(tofu,bowl),2) occurs(cut,knife,tofu,2)
% holds(in(tofu,bowl),3) holds(in(tofu,bowl),4)
Appendix 4 - ASP code of Section 5

time(1..300).
action(grasp1;grasp2;grasp3;align;draw).

occurs(grasp1, lefthand, plank, 50, 85).
occurs(grasp2, lefthand, ruler, 95, 280).
occurs(align, ruler, plank, 100, 168).
occurs(grasp3, righthand, pen, 130, 260).
occurs(draw, pen, plank, 170, 225).

used(X, A1, T1, T2) :- time(T1;T2), action(A1), occurs(A1, X, Y, T1, T2).
used(Y, A1, T1, T2) :- time(T1;T2), action(A2), occurs(A1, X, Y, T1, T2).
used(H, A1, T1, T2) :- time(T1;T2;T3;T4), action(A1;A2),
used(X, A1, T1, T2), used(H, A2, T3, T4),
used(X, A2, T3, T4), T3 < T1, T2 < T4.

% % % Is the ruler aligned when the pen is drawing on the plank?
start(grasping, T1) :- time(T1;T2), occurs(grasp1, X, Y, T1, T2).
end(grasping, T2) :- time(T1;T2), occurs(grasp1, X, Y, T1, T2).
start(grasping, T1) :- time(T1;T2), occurs(grasp2, X, Y, T1, T2).
end(grasping, T2) :- time(T1;T2), occurs(grasp2, X, Y, T1, T2).
start(grasping, T1) :- time(T1;T2), occurs(grasp3, X, Y, T1, T2).
end(grasping, T2) :- time(T1;T2), occurs(grasp3, X, Y, T1, T2).
start(aligning, T1) :- time(T1;T2), occurs(align, X, Y, T1, T2).
end(aligning, T2) :- time(T1;T2), occurs(align, X, Y, T1, T2).
start(drawing, T1) :- time(T1;T2), occurs(draw, X, Y, T1, T2).
end(drawing, T2) :- time(T1;T2), occurs(draw, X, Y, T1, T2).
holds(aligned, T2+1) :- time(T1;T2), occurs(align, X, Y, T1, T2).
holds(drawn, T2+1) :- time(T1;T2), occurs(draw, X, Y, T1, T2).
holds(F, T+1) :- time(T), holds(F, T), not nholds(F, T+1).
nholds(F, T+1) :- time(T), nholds(F, T), not holds(F, T+1).
no :- time(T1;T2;T), start(drawing, T1), end(drawing, T2),
        T1 < T, T < T2, not holds(aligned, T).
yes :- not no.

#show yes/0.
#show used/4.
Appendix 5 - ASP code of Section 6.1

Произведение 5 - ASP код раздела 6.1

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Domain Predicates
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
time(0..300).
event(grasp1;grasp2;align;grasp3;draw).

using(grasper1;grasper2;grasper3;ruler;pen).
dest(plank;ruler;pen).

occurs(grasp1,lefthand,plank,50,85).
occurs(grasp2,lefthand,ruler,95,280).
occurs(align,ruler,plank,100,168).
occurs(grasp3,righthand,pen,130,260).
occurs(draw,pen,plank,170,225).

component(grasp1,grasper1,plank,mark).
component(grasp2,grasper1,ruler,mark).
component(grasp3,grasper2,pen,mark).
component(align,ruler,plank,mark).
before(grasp1,grasp2).
subinterval(align,grasp2).
subinterval(grasp3,grasp2).
subinterval(draw,grasp3).
startsb4ov(align,grasp3).
neq(grasper1,grasper2).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Type Predicates to encode superclass
%%%%% information
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
type(lefthand,grasper1).
type(righthand,grasper1).
type(lefthand,grasper2).
type(righthand,grasper2).
type(lefthand,grasper3).
type(righthand,grasper3).
type(ruler,ruler).
type(pen,pen).

satisfy(activity) :- not donotsatisfy(activity).

donotsatisfy(activity) :- event(X), using(Y), dest(Z),
component(X,Y,Z,A), not occuract(X).

occuract(X) :- event(X), using(Y), dest(Z),
component(X,Y,Z,A), occuract(X).

start(A,X) :- event(A), time(X;Y),
dest(V), using(U1), type(U,U1),
occuract(A,X,V,U).

donotsatisfy(activity) :- event(A1;A2), time(X;Y),
before(A1,A2), end(A1,X),
start(A2,Y), X >= Y.

5
#show satisfy/1.
#show occuract/1.
Appendix 5 - ASP code of Section 6.2

num(1..40).
event(g1;g2;g3;align;draw).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% ENUMERATION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Enumeration over all before, startsb4ov and subinterval predicates
0 { before(A,B) } 1 :- event(A), event(B), A!=B.
0 { startsb4ov(A,B) } 1 :- event(A), event(B), A!=B.
0 { subinterval(A,B) } 1 :- event(A), event(B), A!=B.

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% DEFINITIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Defining pbefore and pstarts4ov in terms of before and starts4ov
pbefore(A,B) :- event(A), event(B), bef(A,B).
pbefore(A,B) :- event(A), event(B), pbefore(A,B).

pstarts4ov(A,B) :- event(A), event(B), starts4ov(A,B).

psubinterval(A,B) :- event(A), event(B), subinterval(A,B).
psubinterval(A,B) :- event(A), event(B), psubinterval(A,B).

bef(A,B) :- event(A), event(B), before(A,B).
bef(A,B) :- event(A), event(B), pbefore(A,B).

subi(A,B) :- event(A), event(B), subinterval(A,B).
subi(A,B) :- event(A), event(B), psubinterval(A,B).

stb4ov(A,B) :- event(A), event(B), starts4ov(A,B).

stb4ov(A,B) :- event(A), event(B), pstarts4ov(A,B).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% CONSTRAINTS(I)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% If A is before B, then it cannot overlap
:- event(A), event(B), pbefore(A,B), pstarts4ov(A,B).
:- event(A), event(B), pbefore(A,B), pstarts4ov(B,A).
:- event(A), event(B), pbefore(A,B), psubinterval(A,B).
:- event(A), event(B), pbefore(A,B), psubinterval(B,A).
:- event(A), event(B), psubinterval(A,B), pstarts4ov(A,B).
:- event(A), event(B), psubinterval(A,B), pstarts4ov(B,A).

% the relations are asymmetric
:- event(A), event(B), pbefore(A,B), pbefore(B,A).
:- event(A), event(B), pstarts4ov(B,A), pstarts4ov(A,B).
:- event(A), event(B), psubinterval(B,A), psubinterval(A,B).

% they are not reflexive
:- event(A), before(A,A).
:- event(A), starts4ov(A,A).
:- event(A), subinterval(A,A).

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% CONSTRAINTS(II)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
:- not pbefore(g1,g2).
:- not pbefore(g1,align).
:- not pbefore(g1,g3).
:- not pbefore(g1,draw).
:- not pbefore(align,draw).

:- not psubinterval(align,g2).
:- not psubinterval(g3,g2).
:- not psubinterval(draw,g3).
:- not psubinterval(draw,g2).
:- not pstarts4ov(align,g3).

%%%%%%%%%%%%%%%%%%%%%% MINIMIZATION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

ecount(N) :- N1 = #count { subinterval(A,B) :
    event(A), event(B), subinterval(A,B)},
   N2 = #count { before(A,B) :
    event(A), event(B), before(A,B)},
   N3 = #count { starts4ov(A,B) :
    event(A), event(B), starts4ov(A,B)},
   N = N1+N2+N3.

#minimize { N@1,ecount : ecount(N)}.

#show before/2.
#show starts4ov/2.
#show subinterval/2.