

Curso-Taller Introducción al Modelamiento de Sistemas Biológicos vía programación lógica ASP (Answer Set Programming)

Centro de Modelamiento Matemático
Blanco Encalada 2120, 7° piso

Orador principal

Santiago Videla (Inria Rennes, Francia)

Answer Set Programming (ASP) es un lenguaje declarativo orientado a modelar y resolver problemas difíciles (NP-hard) de búsqueda y optimización combinatorial.

En este curso-taller introduciremos los conceptos básicos de este lenguaje y presentaremos ejemplos de su aplicación a problemas reales en redes biológicas.

8 al 29 de Noviembre, 2013

7 sesiones

(Lunes y Viernes de 17:00 a 18:30 hrs.)

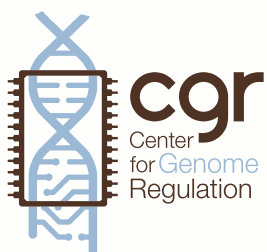
Inscripciones y mayores detalles en:

<http://eventos.cmm.uchile.cl/bioasp/>

Organiza:

Patrocinan:

CMM
Center for
Mathematical
Modeling



fcfm

Ingeniería Matemática
FACULTAD DE CIENCIAS
FÍSICAS Y MATEMÁTICAS
UNIVERSIDAD DE CHILE

Inria
informatics mathematics

```
% external data:
% upstream(MOTIF,GENE,P,Q) :-
P and q-value Q
% inOperon(OPERON,GENE) :- GEN
% binds(GENE,MOTIF,I) :- gene
% mi(G1,G2,MI) :- genes G1 and

% these constants are used to
#const minMI = 0.
#const minIdent = 50.

% if we want to
bindsOp(OPERON,MOTIF) :- inOp
canBind(GENE,MOTIF,IDENT), IDE
upstrOp(MOTIF,OPERON,Q) :- inO
miOp(OA,OB) :- inOperon(OA,GA)

% MOTIF controls OPERON
controls1(MOTIF,OPERON) :- ups
controls2(MOTIF,OPERON) :- not
bindsOp(OH,MM), controls1(MM,
controls3(MOTIF,OPERON) :- not
controls2(MOTIF,OPERON),
upstrOp(MOTIF,OH,Q), bindsOp(
controls4(MOTIF,OPERON) :- not
controls2(MOTIF,OPERON),
not controls3(MOTIF,OPERON),
bindsOp(OH,MM), controls3(MM,
controls5(MOTIF,OPERON) :- not
controls2(MOTIF,OPERON),
not controls3(MOTIF,OPERON),
upstrOp(MOTIF,OH,Q), bindsOp(
controls(MOTIF,OPERON) :- cont
controls(MOTIF,OPERON) :- cont
controls(MOTIF,OPERON) :- cont
controls(MOTIF,OPERON) :- cont

% MOTIF is an explanation of o
explained(A,B) :- miOp(A,B), c
% predicates for counting atom
count_bindsOp(X) :- X = #count
count_explained(X) :- X = #cou
count_controls(X) :- X = #coun
count_miOp(X) :- X = #count{mi

% a non-trivial mutual-inform
ntMI(A,B) :- miOp(A,B), not e
controlled(OPERON) :- controls(
free(OPERON) :- bindsOp(OPERON
free(OPERON) :- miOp(OPERON,_
free(OPERON) :- miOp(_,OPERON)
#hide.
#show ntMI/2.
#show count_bindsOp/1.
#show count_explained/1.
#show count_controls/1.
#show count_miOp/1.
#show free/1.
```