Cross-domain Heuristic Search Challenge:
GISS Algorithm presentation

Alberto Acuña, Victor Parada, and Gustavo Gatica

Universidad de Santiago de Chile,
Av. Libertador Bernardo O’Higgins 3363.
Estación Central, Santiago, Chile
karim@computer.org, {victor.parada, gustavo.gatica}@usach.cl
http://www.usach.cl

Abstract. Generic Iterative Simulated-Annealing Search (GISS) is a
hyper-heuristic algorithm inspired on a Simulated Annealing driven method,
on this context its used to explore across the different instance-heuristic-
state search space. On this extended abstract we describe the proposed
approach, characteristics and others technicals parameters.

Keywords: Simulated Annealing, Hyper-Heuristic, CHeSC

1 Proposed Approach

GISS is a hyper-heuristic algorithm that works in a time driven constructive
way, building iteratively a better solution by each heuristic execution, using a
thermodynamical function to escape from local optima, in relation of the time
limit.

1.1 Characteristics of Heuristic Selection and Execution

GISS heuristic selection use a random number generator to determine witch
heuristic is going to be executed. The temperature function probabilistically
decides between changing or not the execution state. If the chosen heuristic
 correspond to a crossover one, its executed between the current and the last
current solution stored in memory.

1.2 Characteristics of Simulated Annealing

The characteristics of simulated annealing uses a exponent descendant probabil-
ity function, that accepts more movements or heuristic executions as more time
left for all the system.

1.3 Restarting & Auto-parameterization

Restart is applied when a certain numbers of executions doesn’t change solution
state. While Auto-parameterization depends of the gain of each state-time transi-
tion, when it reaches a certain level, deep of search and mutation are increased.
1.4 Pseudo Code

Here we show the pseudo-code of the current approach.

Algorithm 1 GISS Algorithm

Require: \texttt{ProblemDomain}
Ensure: \texttt{exists(ProblemDomain)}
\begin{verbatim}
initSol \{Set initial solution\}
while time < timeMax do
    newSol \leftarrow execHeuristic(i) \{Pic a heuristic\}
    calTtemperature()
    if \texttt{P(newSol, temp(time/timeMax))} > \texttt{random()} then
        initSol = newSol \{Yes, change state\}
    else
        resetCounter ++ \{No, increase reset counter\}
    end if
    if resetCounter > limitCounter then
        resetSystem()
    end if
    if increaseCondition() then
        increase\texttt{(deepSearch)}
        increase\texttt{(deepMutation)}
    end if
end while
\end{verbatim}