Example Session GA32

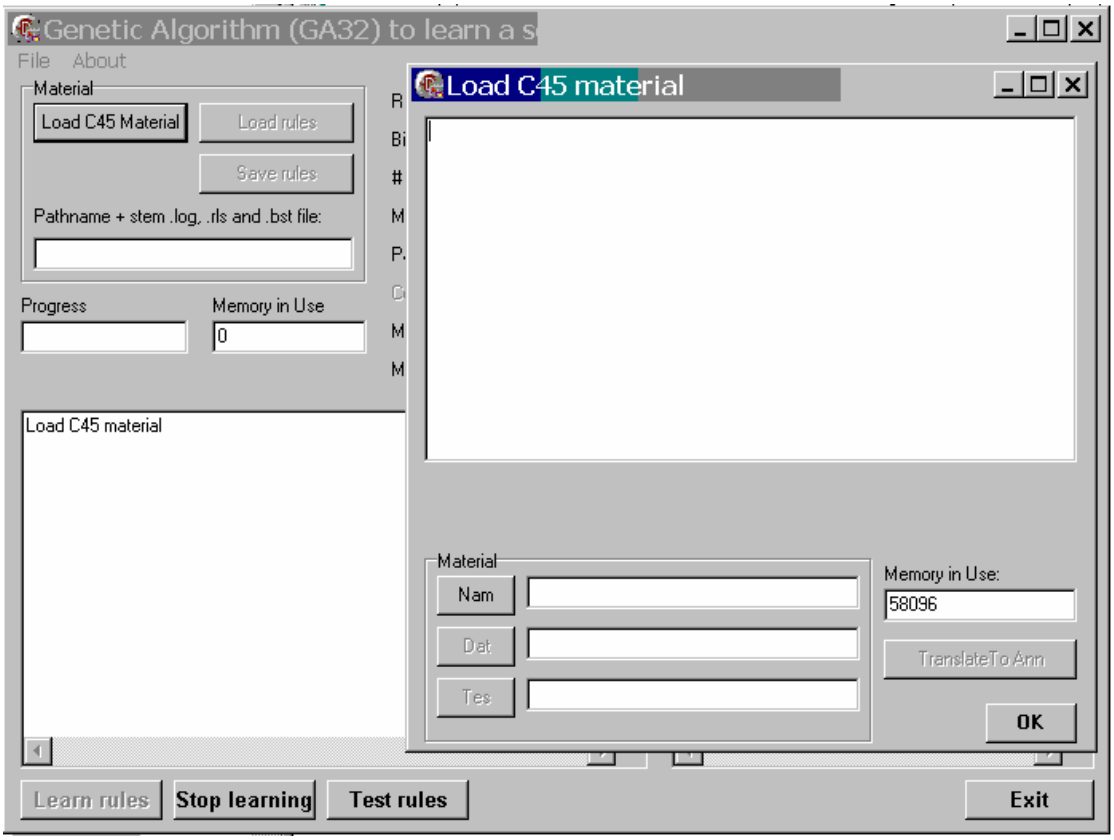
Ton Weijters

Example Session GA32

GA32 is a Genetic Rule Induction Algorithm[1]. The expected input is in the so called C45 format (used by Quinland in one of the first rule induction programs C4.5). For a learning and test session three files are relevant: one file with extension .NAM one with extension .DAT and one with the extension .TES. In this example session we will use the hospital day's prediction example and the following three data sets: PATIENTS.NAM, PATIENTS.DAT and PATIENTS.TES. The file PATIENTS.NAM contains the data definition:

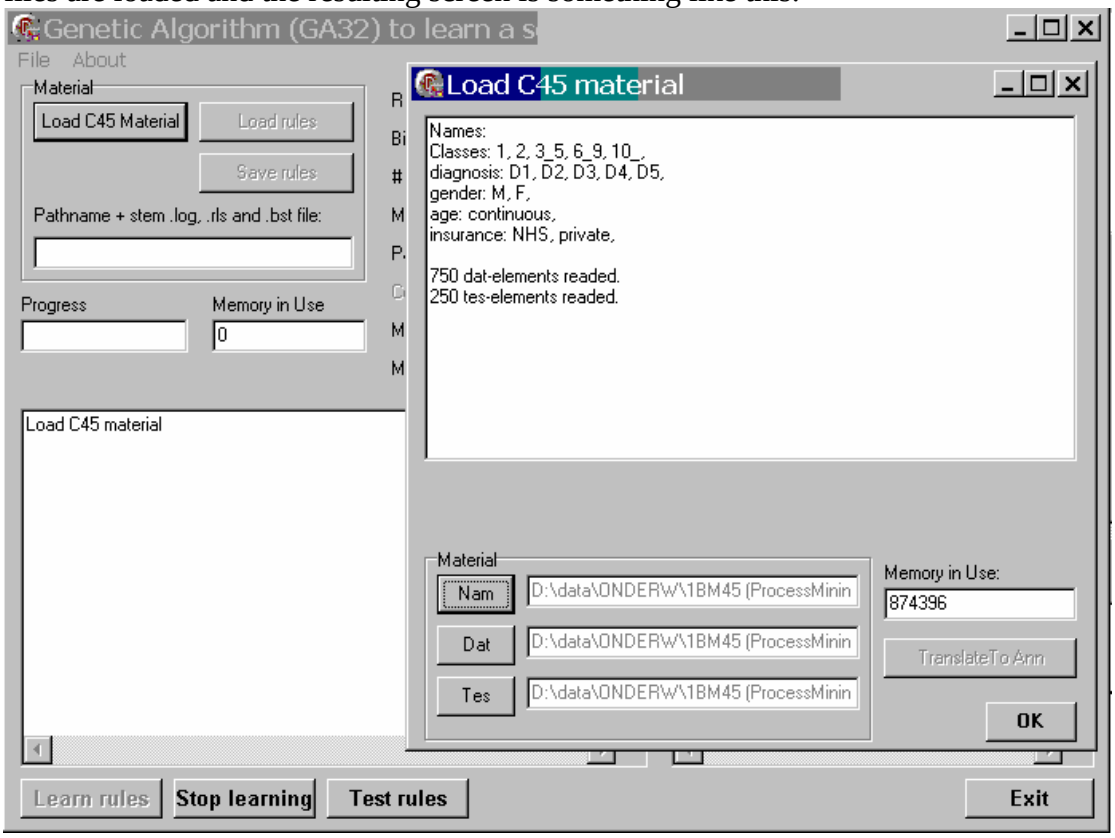
1, 2, 3_5, 6_9, 10_.
diagnosis: D1, D2, D3, D4, D5.
gender: M, F.
age: continuous, 0, 100, 10.
insurance: NHS, private.

The first line contains the possible classifications. Because GA32 can only handle discrete variables, and 'age' is a continuous variable from 0 to 100 we indicate that we translate the continuous values in 10 discrete categories: from 0 to 9 is the first category, 10 to 19 the next, etc. (if you prefer 20 categories change the 10 to 20) . Below we will start an example rule learning session on the PATIENTS data sets. First start GA32 and click the Load C45 Material Button. The result is something like the screen shot below:

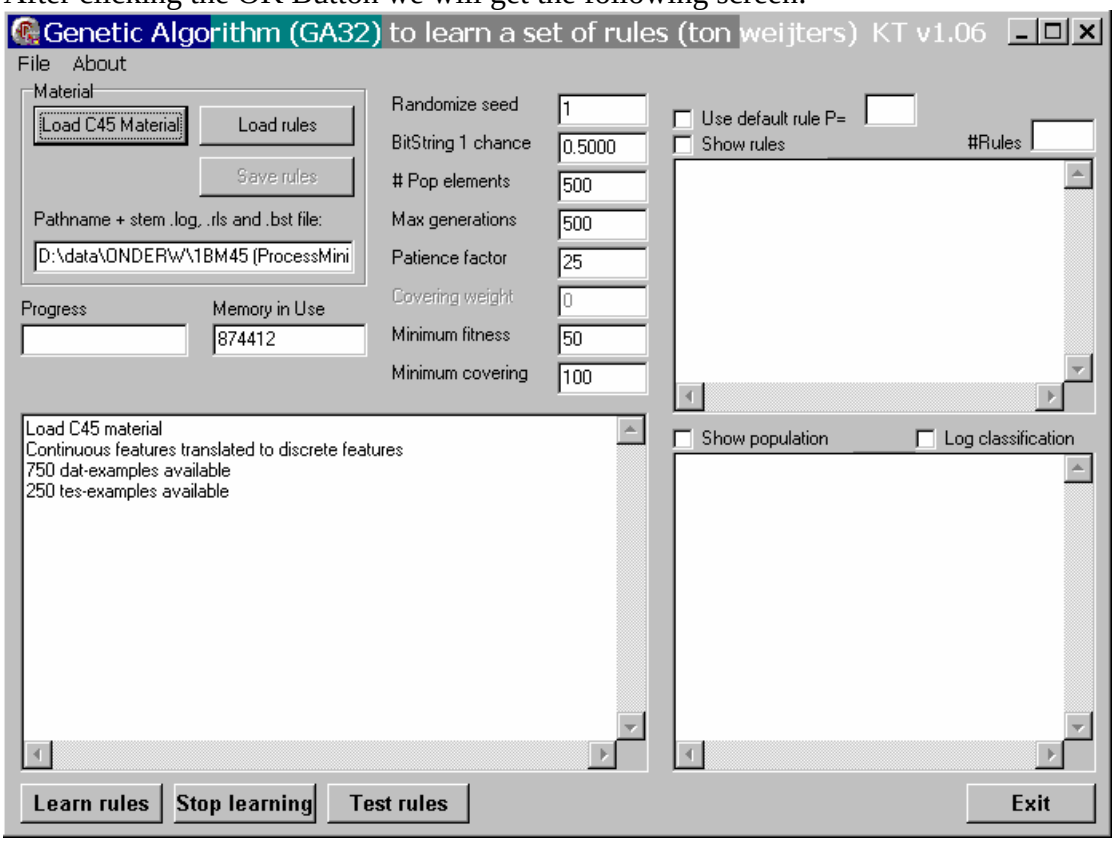


[1] For a motivation and details of the algorithm see Ton Weijters and Jan Paredis (2002). Genetic Rule Induction at an Intermediate Level. *Knowledge-Based Systems*, Vol 15/1-2, pp 85-94.

Click the Nam Button and search for a file PATIENT.NAM. If you load this file and the two other PATIENT-files (DAT and TES) are in the same directory al the three files are loaded and the resulting screen is something like this:



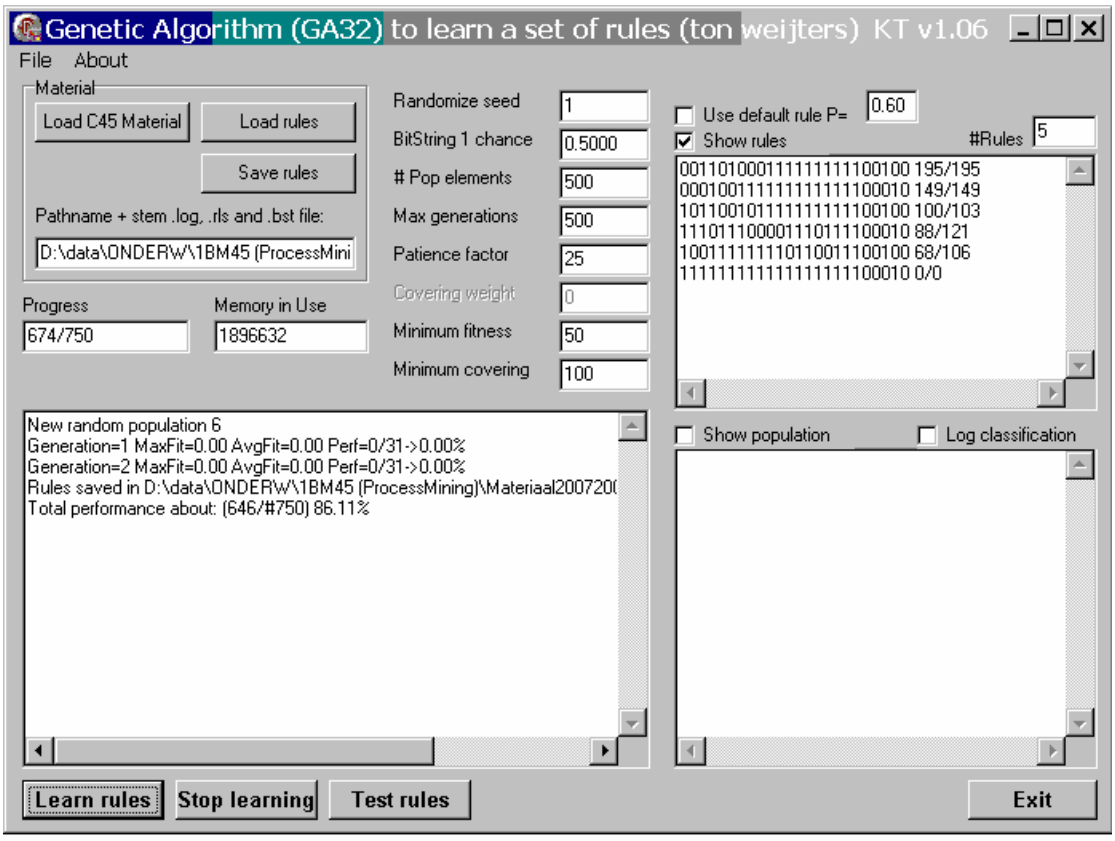
After clicking the OK Button we will get the following screen:



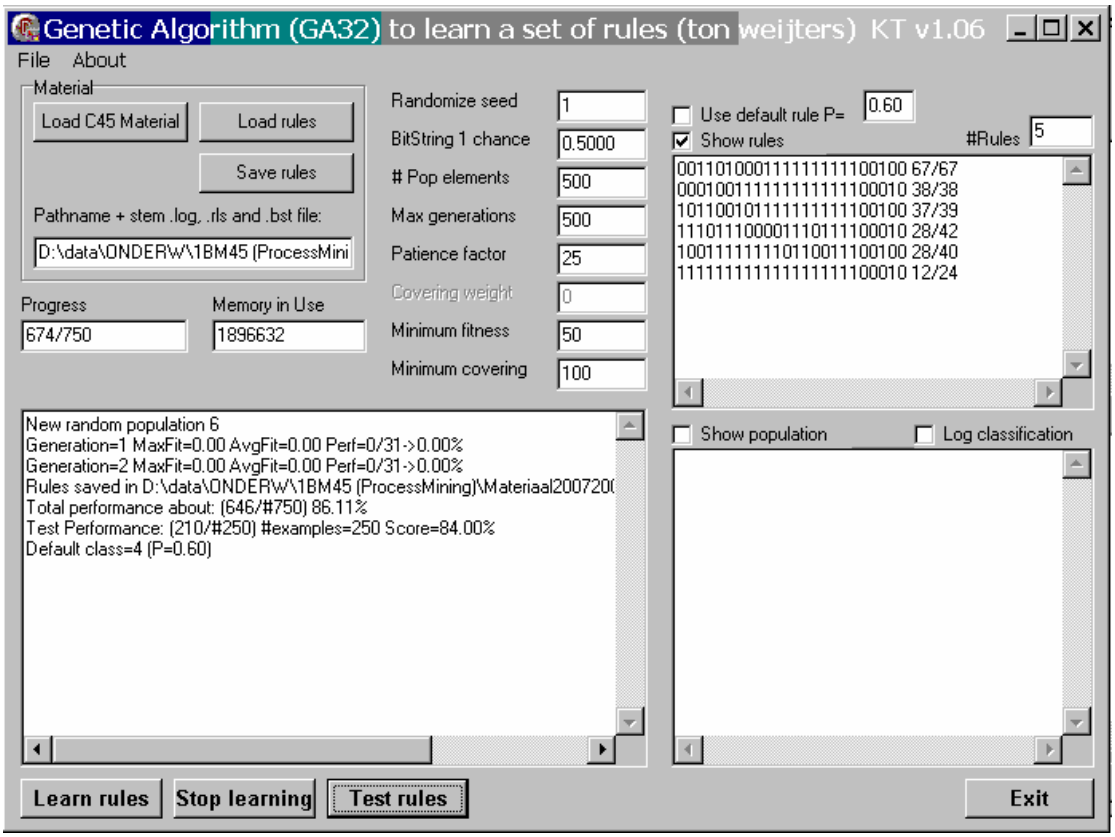
Now it is possibble to change the default parameter setting and to start the learning process. The following parameters are available:

- **Randomize seed** influence the random initialization of the population. An other seed will result in an other start population and possible in an other result.
- **Bit String 1** chance influence the chance that during initialization the vale in the rule representation bit string is 1. In some exceptional situation it can be necessary to change this value.
- **# Pop elements** indicates the number of elements in the population.
- **Max generations** indicates the max number of generations. However, if there is no improvement during *n* generations (*n* is the **Patience factor**) the process will also stop.
- **Patience factor** see under **Max generations**.
- **Minimum fitness** indicates that if the fitness of the best found rule is below this value, the rule is not accepted and the rule searching process stops.
- **Minimum covering** indicates that if the covering of the best found rule is below this value, the rule is not accepted and the rule searching process stops.

By using the different check boxes it is possible to influence the logged information and how much information is showed on the screen. In this example session we only check the show rule box and then we start “Learn rules”. The result is something like this:



The result is a rule set with 5 normal rules plus a so called default rule (rule 6). Because the minimum covering is 100 and the minimum fitness is 50 all rules satisfy this. Rule 5 covers 106 cases and the classification of 68 cases is correct. The information about the covering and correct classification of the default rule is not calculated but the classification performance seems 60% (P=0.60). By using the “Test rules” button we can test the performance of the rules on the test material. Remark that after testing the rules, the performance of the rules on the test material is displayed.



Beside the information on the screen, the whole learning session is logged in the file PATIENTS.LOG. The file contains information about the parameter settings, the rules in a binary representation, coverring and classification information on learning and test material. The log file also contains a translation of the binary rules to a more user friendly format. Finally, the classification matrix is given. Each new GA32 sessions is appended to the log file. If you like to start a new logging rename or remove the old file.

If we check the “Log classification” box, the log file also contains information about the individual classifications of the learning and test examples.

Warning: Do not try to restart the learning process with an other parameter setting. Exit the program and start a new process!

```
============================================================
Genetic Algorithm (GA32)
12/02/2008  15:45:25
============================================================
Fraction class 1: 0.0053
Fraction class 2: 0.0080
Fraction class 3: 0.5200
Fraction class 4: 0.4000
Fraction class 5: 0.0670
Continuous features translated to discrete features
750 dat-examples available
250 tes-examples available
Stem of the training and test data: D:\data\ONDERW\1BM45
(ProcessMining)\Materiaal20072008\Assignment2\PATIENTS
UseDefault = FALSE class=3 P=0.5200
Seed = 1
BitString1Chance: 0.5000
Number of rules in the population: 500
Maximum number of generations: 500
Next generation if max fitness is # times equal: 25
Covering Weight: 0.0000
RuleReliabilityThres <: 50.0000
Minimum Covering: 100
Start: 12/02/2008  16:07:59
New random population 1
Generation 72
R1: 0011010001111111111100100 OK=195 Match=195
Default class=4 (P=0.54)
New random population 2
Generation 68
R2: 0001001111111111111100010 OK=149 Match=149
Default class=3 (P=0.48)
New random population 3
Generation 66
R3: 1011001011111111111100100 OK=100 Match=103
Default class=4 (P=0.50)
New random population 4
Generation 66
R4: 1110111000011101111100010 OK=88 Match=121
Default class=3 (P=0.43)
New random population 5
Generation 65
R5: 1001111111110110011100100 OK=68 Match=106
Default class=4 (P=0.60)
New random population 6
New random population 6
New random population 6

Name representation of the rules:
================================

IF                          (R1 195/195)
diagnosis in {D3 D4 }
gender=M
age in
[20..30][30..40][40..50][50..60][60..70][70..80][80..90][90..100]
THEN
class = 3_5

IF                          (R2 149/149)
diagnosis=D4
```

```
gender=F
THEN
class = 6_9

IF                          (R3 100/103)
diagnosis in {D1 D3 D4 }
gender=F
age in
[10..20][20..30][30..40][40..50][50..60][60..70][70..80][80..90][90..
100]
THEN
class = 3_5

IF                          (R4 88/121)
diagnosis in {D1 D2 D3 D5 }
age in [40..50][50..60][60..70][80..90][90..100]
THEN
class = 6_9

IF                          (R5 68/106)
diagnosis in {D1 D4 D5 }
age in [0..10][10..20][20..30][30..40][50..60][60..70][90..100]
THEN
class = 3_5

IF                          (R6 0/0)
THEN
class = 6_9

Total performance about: (646/#750) 86.11%
Ready: 12/02/2008  16:08:16
Test Performance: (210/#250) #examples=250 Score=84.00%

Confusion Matrix:
      |     1     2     3     4     5 << Target class
-------------------------------------
     1|     0     0     0     0     0
     2|     0     0     0     0     0
     3|     1     0   132     4     9
     4|     3     0     8    78    15
     5|     0     0     0     0     0
Current
classification

R1: 00110100011111111100100 OK=67 Match=67
R2: 00010011111111111100010 OK=38 Match=38
R3: 10110010111111111100100 OK=37 Match=39
R4: 11101110000111011100010 OK=28 Match=42
R5: 10011111111010011100100 OK=28 Match=40
R6: 11111111111111111100010 OK=12 Match=24
Default class=4 (P=0.60)
```