2011

Large Scale Data Mining using Genetics-Based Machine Learning

Jaume Bacardit

School of Computer Science &
School of Bioscience
University Nottingham
Nottingham, UK

jqb@cs.nott.ac.uk

Xavier Llorà

Data-Intensive Technologies and Applications
National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign
Urbana, IL 61801

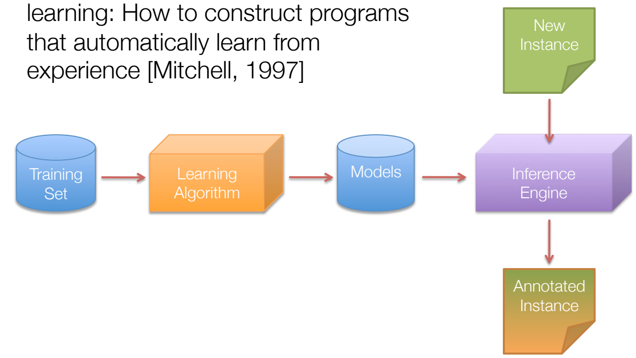
xllora@illinois.edu

GECCO 2011 Tutorial, July 12th 2011

GECCO'11, July 12–16, 2011, Dublin, Ireland.
ACM 978-1-4503-0690-4/11/07.

Machine Learning and Data Mining

- Core of Data Mining ➔ Machine learning: How to construct programs that automatically learn from experience [Mitchell, 1997]



What Will We Cover?

- What does large scale mean?
- Evolution as massive parallel processing
- The challenges of data mining
- Kaleidoscopic large scale data mining
- Real examples
- Summary and further directions

WHAT DOES LARGE SCALE MEAN?

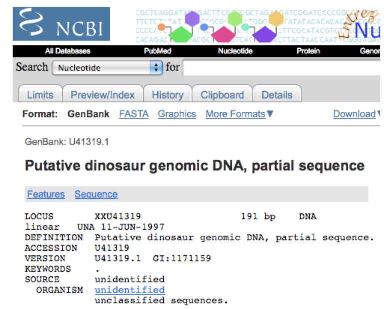
Evolution as massive parallel processing
The challenges of data mining
Kaleidoscopic large scale data mining
Real-world examples
Wrapping up

What Does Large Scale Mean?

- Many scientific disciplines are currently experiencing a massive “data deluge”
- Vast amounts of data are available thanks to initiatives such as the human genome project or the virtual human physiome
- Data mining technologies need to deal with large volumes of data, scale accordingly, extract accurate models, and provide new insight
- So, what does large mean?

Large Meaning… Piles of Records

- Datasets with a high number of records
  - This is probably the most visible dimension of large scale data mining
  - GenBank (the genetic sequences database from the NIH) contains (Feb, 2008) more than 82 million gene sequences and more than 85 billion nucleotides



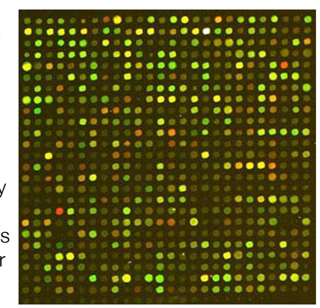
Large Meaning… Piles of Records

- Datasets with a high number of records
  - Not all data comes from the natural sciences
  - Netflix Prize:
    - Generating better movie recommending methods from customer ratings
    - Training set of 100M ratings from over 480K customers on 78K movies
    - Data collected from October 1998 and December, 2005
    - Competition lasted from 2006 to 2009

- Think big: Twitter, Facebook?

Large Meaning… High Dimensionality

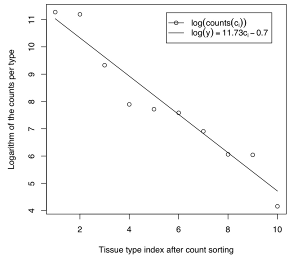
- High dimensionality domains
  - Sometimes each record is characterized by hundreds, thousands (or even more) features
  - Microarray technology (as many other post-genomic data generation techniques) can routinely generate records with tens of thousands of variables
  - Creating each record is usually very costly, so datasets tend to have a very small number of records. This unbalance between number of records and number of variables is yet another challenge



(Reinke, 2006, Image licensed under Creative Commons)

Large Meaning… Rare

- Class unbalance
  - Challenge to generate accurate classification models where not all classes are equally represented
  - Contact Map prediction datasets (briefly explained later in the tutorial) routinely contain millions of instances from which less than 2% are positive examples
  - Tissue type identification is highly unbalance—see figure



(Llora, Priya, Bhargava, 2009)

Large Meaning… Lots of Classes

- Yet another dimension of difficulty
- Reuters-21578 dataset is a text categorization task with 672 categories
- Very related to the class unbalance problem
- Machine learning methods need to make an extra effort to make sure that underrepresented data is taken into account properly

Large Meaning… Complex Concepts

- Bernado and Ho (2005) propose complexity measures for classification tasks
- Metrics to measure difficulty in classifiers
  - 9 different ones studied
  - Geometry
  - Sparseness
  - Dimensionality
  - Sample size
- Metric creation is a hard problem since dimensions of complexity may be intertwined

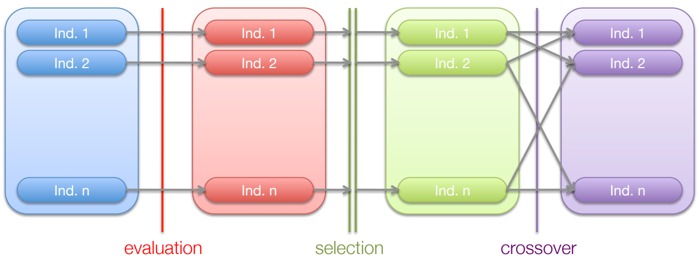
What does large scale mean?

EVOLUTION AS MASSIVE PARALLEL PROCESSING

The challenges of data mining
Kaleidoscopic large scale data mining
Real-world examples
Wrapping up

Evolution and Parallelism

- Evolutionary algorithms are parallelism rich
- A population is data rich (individuals)
- Genetic operators are highly parallel operations



Operations and Their Dependencies

- No dependencies ➔ embarrassing parallelism
  - Fitness evaluation
  - Each individual can be evaluated simultaneously
- Weak dependencies ➔ synchronization points
  - Crossover
  - Once the parents are available the operator can be applied
- Strong dependencies ➔ careful inspection (bottlenecks)
  - Selection
  - The complete population needs to be available
  - The wrong implementation can introduce large serial execution chunks

Other Perks

- Need to repeat experiments
- Evaluation can be costly
- Some evolutionary models
  - Mimic natural evolution introducing spatial relations (remember Darwin's islands?)
  - Are model after decentralized models (cellular automata like)
- Based on the nature of evolutionary algorithms and the above ingredients there multiple parallelization models has been proposed (Cantu-Paz, 2000; Alba, 2005)

But?

- What about the data?

What does large scale mean?
Evolution as massive parallel processing

THE CHALLENGES OF DATA MINING

Kaleidoscopic large scale data mining
Real-world examples
Wrapping up

The Challenges of Data Mining

- We have seen in the previous slides how evolutionary algorithms have a natural tendency for parallel processing, hence being suitable for large-scale data mining
- However, data mining presents a challenge that goes beyond pure optimization, which is that evaluation is based on *data*, not just on a fitness formula

The Challenges of Data Mining

- Holding the data is the first bottleneck that large-scale data mining needs to face
  - Efficiently parsing the data
  - Proper data structures to achieve the minimum memory footprint
    - It may sound like just a matter of programming, but it can make a difference
    - Specially important when using specialized hardware (e.g. CUDA)
  - Optimized publicly available data handling libraries exist (e.g. the HDF5 library)

The Challenges of Data Mining

- Usually it is not possible to hold all the training data in memory
  - Partition it and use different subsets of data at a time
    - Windowing mechanisms, we will talk about them later
    - Efficient strategies of use of CUDA technology
  - Hold different parts of the data in different machines
    - Parallel processing, we will also talk about this later
- Can also data richness become a benefit not a problem?
  - Data-intensive computing

The Challenges of Data Mining

- Classic challenges of machine learning
  - Over-learning
    - Our models need to have good *predictive* capacity
  - Generating interpretable solution
    - Discovering *useful* new knowledge inside the data

KALEIDOSCOPIC LARGE SCALE DATA MINING

Large Scale Data Mining Using GBML

- Efficiency enhancement techniques
- Hardware acceleration techniques
- Parallelization models
- Data-intensive computing

Prelude: Efficiency Enhancement

- Review of methods and techniques explicitly designed for data mining purposes
- Evolutionary computation efficiency enhancement techniques could also be applied (and we show some examples of this too)
- For a good tutorial on efficiency enhancement methods, please see GECCO 2005 Tutorial on efficiency enhancement by Kumara Sastry at
  - http://www.slideshare.net/kknsastry/principled-efficiency-enhancement-techniques

Efficiency Enhancement Techniques

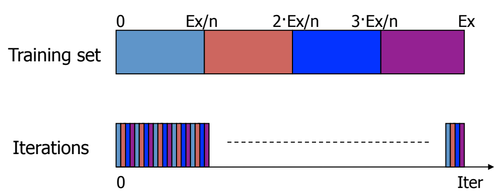
- Goal: Modify the data mining methods to improve their efficiency without special/parallel hardware
- Remember:
  - An individual can be a rule, or a rule set, or a decision tree…
  - Individuals parameters need to be estimated (accuracy, generality…)
- Included in this category are:
  - Windowing mechanisms
  - Exploiting regularities in the data
  - Fitness surrogates
  - Hybrid methods

Windowing Mechanisms

- Classic machine learning concept
  - Do we need to use all the training data all the time?
  - Using a subset would result in faster evaluations
  - How do we select this subset and how often is it changed?
  - How accurate the fitness estimation will be? Will it favor modularity?
- Freitas (2002) proposed a classification of these methods in three types:
  - Individual-wise: Changing the subset of data for each evaluated solution
  - Generation-wise: Changing the subset of data at each generation of the evolutionary algorithm
  - Run-wise: Selecting a single subset of data for a whole run of a GA

Windowing Mechanisms - ILAS

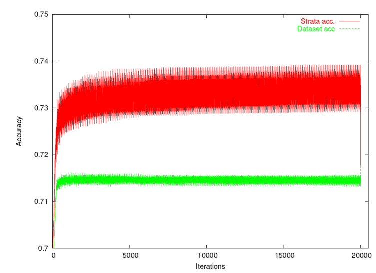
- Incrementing Learning with Alternating Strata (Bacardit, 2004)
- Generation-wise windowing mechanism
- Training set is divided in non-overlapping strata
- Each GA iteration uses a different strata, using a round-robin policy (evaluation speedup linearly with the number of strata)



- This mechanism also introduces some extra generalization pressure, since good solutions need to survive multiple strata

Windowing Mechanisms - ILAS

- How far can we increase the number of strata?
- Problem with ~260K instances and 150 strata
- Knowledge learnt on different strata does not integrate successfully into a single solution (if too many are used)
- We have to make sure that each strata is a good representation of the overall training set
- Success model of the number of strata (Bacardit et al., 2004)



$$P(success/s) = e^{-rs \cdot e^{-\frac{pD}{s}}}$$

r = #rules in solution, s = #strata,
p = prob. rule represented in strata,
D = size of the training set

Exploiting Regularities

- The instances in the training set do not usually cover uniformly the search space
- Instead, there are some recurrent patterns and regularities, that can be exploited for efficiency purposes
- (Giraldez et al., 2005) proposed a method that precomputes the possible classifications of a rule
- As they only dealt with discrete/discretized attributes, they generate a tree structure to efficiently know which examples belong to each value of each attribute
- Furthermore, rule matches are the intersection of all these subsets of examples

Exploiting Regularities in the Data

- Other methods exploit a different regularity: usually not all attributes are equally important
- Example: Prediction of a Bioinformatics dataset (Bacardit and Krasnogor, 2009)
  - Att $Leu_{-2} \in [-0.51,7]$ and $Glu \in [0.19,8]$ and $Asp_{+1} \in [-5.01,2.67]$ and $Met_{+1} \in [-3.98,10]$ and $Pro_{+2} \in [-7,-4.02]$ and $Pro_{+3} \in [-7,-1.89]$ and $Trp_{+3} \in [-8,13]$ and $Glu_{+4} \in [0.70,5.52]$ and $Lys_{+4} \in [-0.43,4.94] \rightarrow$ alpha
  - Only 9 attributes out of 300 were actually in the rule

Exploiting Regularities in the Data

- Function match (instance x, rule r)

```
Foreach attribute att in the domain
    If att is relevant in rule r and
      (x.att < r.att.lower or x.att > r.att.upper)
        Return false
    EndIf
EndFor
Return true
```

- Given the previous example of a rule, 293 iterations of this loop are wasted !!

Exploiting Regularities in the Data

- How to exploit this phenomenon?
- Reordering the attributes in the domain from specific to general (Butz et al., 2008)
  - Afterwards, starting the match process with the most specific one
  - The most specific attributes are usually those that make the process break. Thus, reducing usually the number of iterations in the match loop
  - Still, in the cases where a whole rule matches, the irrelevant attributes need to be evaluated

Exploiting Regularities in the Data

- Could we completely get rid of the irrelevant attributes?
  - The attribute list knowledge representation (ALKR) (Bacardit, Burke and Krasnogor, 2009)
  - This representation *automatically identifies* which are the relevant/specific attributes for each rule
  - Only tracks information about them

#Expr. Atts.	4
Expr. Atts.	1 | 3 | 4 | 7
Intervals	L_1 | U_1 | L_3 | U_3 | L_4 | U_4 | L_7 | U_7
Class	C_1

Exploiting Regularities in the Data

- In ALKR two operators (specialize and generalize) add or remove attributes from the list with a given probability, hence exploring the *rule-wise* space of the relevant attributes
- ALKR match process is more efficient, however crossover is costlier and it has two extra operators
- Since ALKR chromosome only contains relevant information, the exploration process is more efficient. On large data sets it managed to generate better solutions

Fitness Surrogates

- In evolutionary algorithms, we can construct a function that *estimates* the evaluation of our solutions using the training set. This is usually known as a *fitness surrogate*
- Two recent works (Orriols et al., 2007) and (Llorà et al., 2007) use the structural information extracted from the model building process of competent genetic algorithms to build such a function
- Cheap surrogates can help avoid costly evaluations that tend to dominate execution time

Hybrid Methods

- The Memetic Pittsburgh Learning Classifier Systems (MPLCS) (Bacardit and Krasnogor, 2009) combines the classic GA exploration operators with local search (LS) methods.
  - The LS operators use information extracted from the evaluation process
  - After evaluating a rule set we know
    - Which rules are good and which rules are bad
    - Which parts of each rule are good and which parts are bad

Hybrid Methods

- Two kinds of LS operators
  - Rule set-wise operator
    - Takes N parents (N can be > 2) and generates a single offspring with the best rules of all of them
  - Rule-wise operators that edit rules
    - Rule cleaning – drop conditions that misclassify
    - Rule splitting – find the exact spot where a rule can be splitted and the generated rules cleaned
    - Rule generalizing –update a rule so it can correctly classify more examples

Enough Talk! Where is the Big Iron?

- Let's start with a simple hardware acceleration example

Hardware Acceleration Techniques

- Commodity hardware provides simple vectorized operations
- Result of the gaming world
- Usually operate over 128 bits (4 floats)
- Vector units are able to execute ops in 1 cycle
- IBM implemented Altivec
- Intel started with MMX and then SSE and derivates
- AMD 3DNow!, 3DNow+!

A Simple Example: XCSlib

- Llora and Sastry (2005) show its usefulness. Also key to billion bit effort by Golberg, Sastry, and Llora (2007)
- XCSlib version 0.34 (http://xcslib.sourceforge.net/)
  - Based on a C++ code base
  - Very flexible to modify/add new component
- The first step: Gather the facts
- Need to get a clear picture of the execution profile
  - Shark freely available on Mac OS X
  - Gprof on Unix systems

XCSlib

- Shark G4 platform profile (same behavior displayed on the AMD platform)
- The rule matching is conducted by `ternary_condition::match`

XCSlib version 0.34

11-input multiplexer		20-input multiplexer	
%	*function*	*%*	*function*
65.4%	ternary_condition::match	69.6%	ternary_condition::match
8.4%	xcs_classifier_system::select_delete_rw	10.2%	xcs_classifier_system::select_delete_rw
7.5%	binary_state::string_value	7.5%	binary_state::string_value
5.7%	experiment_mgr::perform_experiments	3.1%	xcs_classifier_system::match
3.8%	xcs_classifier_system::match	2.7%	experiment_mgr::perform_experiments
0.9%	xcs_random::dice	1.0%	xcs_classifier_system::update_fitness
0.9%	multiplexer_env::begin_problem	0.7%	action_base<boolean_action>::operator==
0.9%	xcs_classifier_system::update_fitness	0.5%	xcs_random::dice
37-input multiplexer		*70-input multiplexer*	
time	*function*	*%*	*function*
78.5%	ternary_condition::match	85.0%	ternary_condition::match
6.5%	xcs_classifier_system::select_delete_rw	6.3%	binary_state::string_value
6.3%	binary_state::string_value	3.1%	xcs_classifier_system::match
3.2%	xcs_classifier_system::match	1.1%	experiment_mgr::perform_experiments
1.4%	experiment_mgr::perform_experiments	0.8%	ternary_condition::~ternary_condition
0.6%	xcs_classifier::match	0.7%	ternary_condition::cover
0.6%	ternary_condition::~ternary_condition	0.6%	xcs_classifier::match
0.4%	ternary_condition::cover	0.5%	ternary_condition::string_value

`ternary_condition::match`

XCSlib

```
bool
ternary_condition::match(const binary_state& sens)
{
     string::size_type bit;
     string input;
     bool result;

     input = sens.string_value();
     assert(input.size()==bitstring.size());

     bit = 0;
     result = true;

     while ( (result) && (bit<bitstring.size()) ) {
        result = ( (bitstring[bit]=='#') ||
                   (bitstring[bit]==input[bit]) );
        bit++;
     }

     return result;
}
```

- The main cycle consumer
- Each rule loops to match
- Good candidate for HW acceleration
- If we accelerate the inner loop we can drop the time spent matching

Extending Toward Vector Instructions

```
int isRuleMatched ( RULE rule, INSTANCE ins )
     register int i,iFlag;

     for ( i=0, iFlag=1 ;
           i<=RECODE_BLOCKS /*&& iFlag*/ ;
           i++)
        if ( (rule[i]&ins[i]) != ins[i] )
           iFlag = 0;

     return iFlag;
}
```

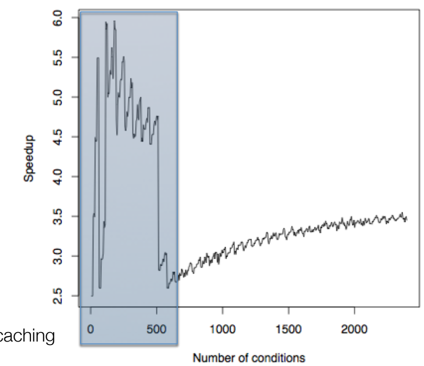
Idea: Loop unroll, using vector operations to manipulate four integers at once (pack 64 conditions in a single match step)

The Vector-based Matching (SSE2)

```
// Matching using SSE2 instruction set
register int i,iMax,tmp;
__m128i vir,vii;

for ( i=0, iMax=RECODE_BLOCKS/4   ;
      i<iMax /*&& iFlag*/ ;
      i++) {
   tmp = i*4;
   vir = _mm_load_si128((__m128i*)&rule[tmp]);
   vii = _mm_load_si128((__m128i*)&ins[tmp]);
   vir = _mm_and_si128(vir,vii);
   vii = _mm_cmpeq_epi32(vir,vii);
   iFlag &= (-1 == _mm_movemask_epi8(vii));
}
```

Speedup After Vectorizing

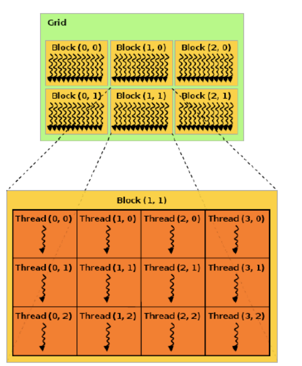


Benefits of caching

Hardware Acceleration On Steroids

- NVIDIA's Computer Unified Device Architecture (CUDA) is a parallel computing architecture that exploits the capacity within NVIDIA's Graphic Processor Units
- CUDA runs thousands of threads at the same time → Single Program, Multiple Data paradigm
- In the last few years GPUs have been extensively used in the evolutionary computation field
  - Many papers and applications are available at http://www.gpgpgpu.com
- The use of GPGPUs in Machine Learning involves a greater challenge because it deals with more data but this also means it is potentially more parallelizable

CUDA architecture



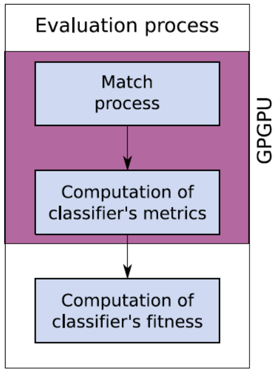
*From NVIDIA CUDA Programming Guide

CUDA memories

- Different types of memory with different access speed
  - Global memory (slow and large)
  - Shared memory (block-wise; fast but quite small)
  - Constant memory (very fast but very small)
- The memory is limited
- The memory copy operations involve a considerable amount of execution time
- Since we are aiming to work with large scale datasets a good strategy to minimize the execution time is based on the memory usage

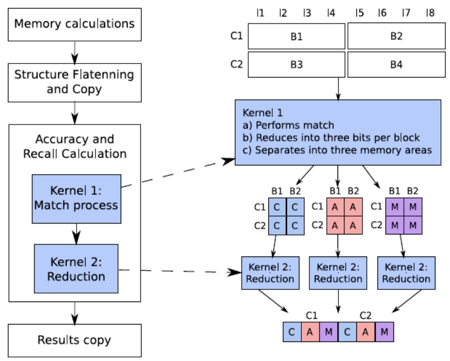
CUDA in supervised learning

- The match process is the stage computationally more expensive
- However, performing only the match inside the GPU means downloading from the card a structure of size O (NxM) (N=population size, M=training set size)
- In most cases we don't need to know the specific matches of a classifier, just how many (**reduce the data**)
- Performing the second stage also inside the GPU allows the system to reduce the memory traffic to O(N)



CUDA fitness computation for the BioHEL GBML system

- BioHEL [Bacardit, Burke and Krasnogor, 2009] is a GBML method designed for large-scale datasets
- We recently extended it with a CUDA-based fitness computation (Franco, Krasnogor & Bacardit, 2010)



Performance of BioHEL using CUDA

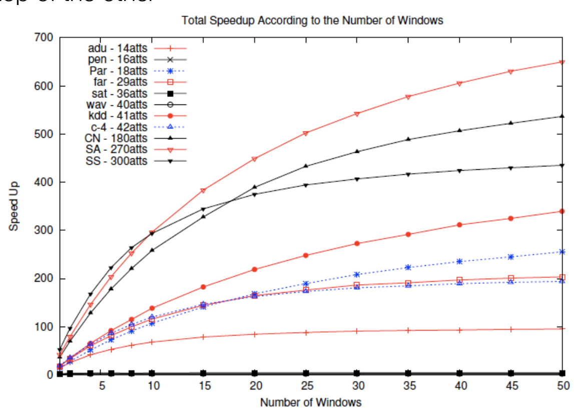
- We used CUDA in a Tesla C1060 card with 4GB of global memory, and compared the run-time to that of Intel Xeon E5472 3.0GHz processors

	Name	|T|	#Att	#Disc	#Cont	#Cl	T. Serial (s)	T.CUDA (s)	Speed Up
Cont.	sat	5790	36	0	36	6	0.03± 0.01	25.91± 2.45	3.7
	wav	4539	40	0	40	3	75.47± 9.38	24.69± 0.81	3.1
	pen	9892	16	0	16	10	149.70± 19.93	40.04± 2.94	3.7
	SS	75583	300	0	300	3	347979.80± 60982.74	5992.28±247.50	58.1
	CN	234638	180	0	180	2	821464.70±167542.04	18644.31±943.98	44.1
Mixed	adu	43960	14	8	6	2	5422.78± 1410.71	271.73± 26.03	20.0
	far	90868	29	24	5	8	2471.28± 701.83	94.99± 41.53	26.0
	kdd	444619	41	15	26	23	76442.32± 23533.21	2102.414±191.34	36.4
	SA	493788	270	26	244	2	1252976.80±203186.55	28759.71±552.00	38.3
	Par	235929	18	18	0	2	524706.70± 98949.46	19559.79±671.70	26.8
	c-4	60803	42	42	0	3	52917.95± 8059.55	2417.83±170.19	21.9

- Biggest speedups obtained in large problems (|T| or #Att), specially in domains with continuous attributes
- Run time for the largest dataset reduced from **2 weeks** to **8 hours**

Integration of CUDA and ILAS

- The speedups of these two techniques can stack one on top of the other



Parallelization Models

- Coarse-grained parallelism
- Fine-grained parallelism

Coarse-grained Parallelism

- By coarse-grain parallelism we are talking about executing independently several runs
- As there is no communication, the speedup is always linear ☺
- In which situations can we do this?
  - Evolutionary algorithms are stochastic methods, we need to run always our methods several times. If we have the parallel hardware, this is a trivial way of gaining efficiency

Coarse-grained Parallelism

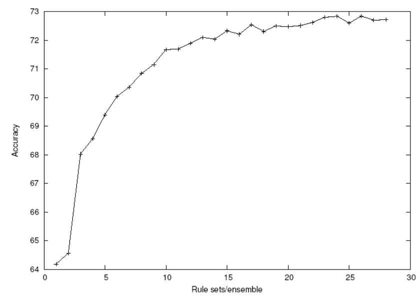
- There is, however, a more defined way of performing coarse-grain parallelism: ensemble learning
- These techniques integrate the collective predictions of a set of models in some principled fashion
- These models can be trained independently

Coarse-grained Parallelism

- Ensemble for consensus prediction (Bacardit and Krasnogor, 2008)
- Similar technique to bagging
  1. Evolutionary data mining method is run N times on the original training set, each of them with a different random seed
  2. From each of the N runs, a rule set is generated
  3. Exploitation stage: For each new instance, the N models produce a prediction. The majority class is used as the ensemble prediction
- Ensembles evaluated on 25 UCI repository datasets using the Gassist LCS
- In average the ensemble accuracy was 2.6% higher

Coarse-grained Parallelism

- Ensemble for consensus prediction
  - Prediction of a difficult bioinformatics dataset
  - Accuracy increased of ~9% with 25 rule sets

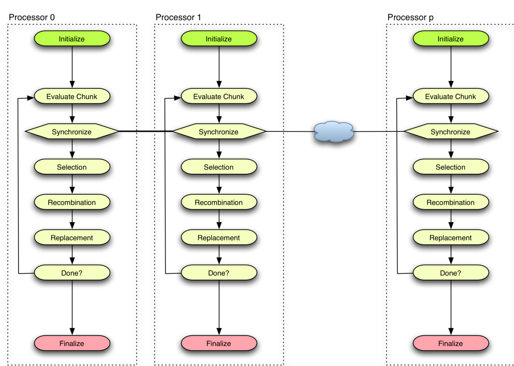


More Corse-Grain

- If evaluation is extremely costly
- Run the same algorithm with the same seed
- Same population everywhere
- Each algorithm only evaluates a chunk of the population
- The fitness estimates are broadcasted (e.g. MPI)
- Minimal communication possible (only the fitness value)
- All algorithms run the same genetic operators on identical population individuals (as all run using the same seed)

- The NAX system (Llora, X., Priya, A., and Bhragava, 2007)

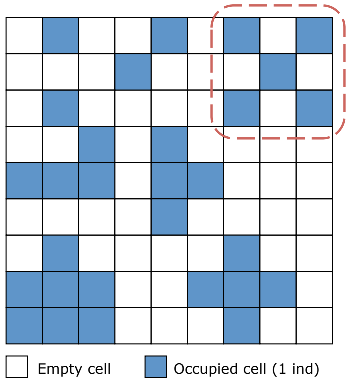
In a Picture

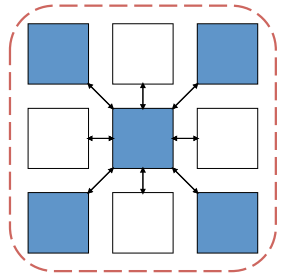


Fine-grained parallelism

- Exploit maximum parallelism
- Impose an spatial topology
- Define neighborhood operators
- GALE (Llora, 2002)
- Easy implementable on shared-memory machines
- Minimizes the computation/communication ratio for distributed memory implementations

GALE: Topology

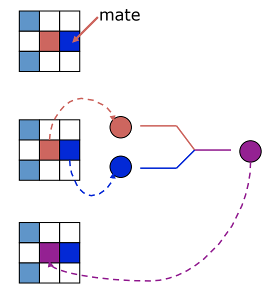




- A cell contains 0 or 1 individual
- A cell is surrounded by 8 neighbors
- Subpopulations spatially defined by the adjacent cells

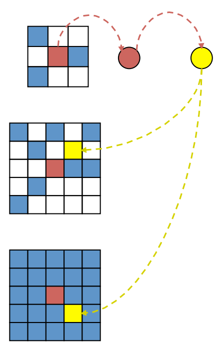
GALE: Merge

- Merge
  1. Choose a neighbor
  2. Recombine the genetic material
  3. Replace the individual

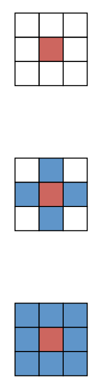


GALE: Split

- Split
  1. Replicate and mutate
  2. Occupy
     - Empty cell with higher number of neighbors
     - Occupied cell with the worst neighbor (no empty cell available)

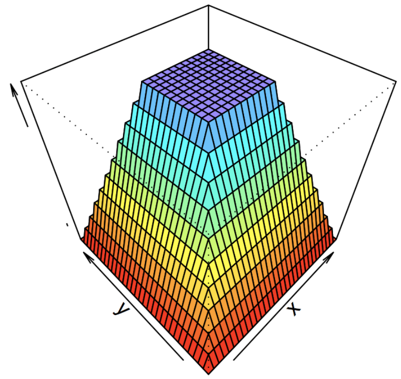


GALE: Survival



- 0-1 Neighbors
  - Isolated
  - $p_{sr}(ind)$ fitness proportional
  - death ➔ leave cell empty
- 2-6 Neighbors
  - Spongy
  - $p_{sr}(ind)$ related to neighbors
  - death ➔ leave cell empty
- 7-8 Neighbors
  - Crowded
  - $p_{sr}(ind)$ = 0
  - death ➔ replace by the best

GALE: Data Distribution



Data-intensive Computing

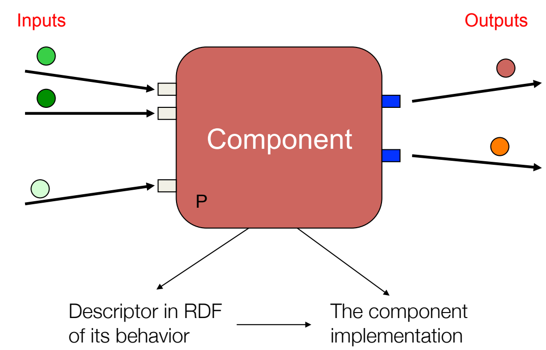
- Usually refers to:
  - Infrastructure
  - Programming techniques/paradigms
- Google made it main stream after their MapReduce model
- Yahoo! provides and open source implementation
  - Hadoop (MapReduce)
  - HDFS (Hadoop distributed filesystem)
  - Mahout (Machine Learning methods)
- Engineered to store petabytes reliably on commodity hardware (fault tolerant)
- Map: Equivalent to the map operation on functional programming
- Reduce: The reduction phase after maps are computed

Meandre: NCSA's Data-Intensive Infrastructure

- Extend the programming limitation of MapReduce
- Execution Paradigms
  - Conventional programs perform computational tasks by executing a sequence of instructions.
  - Data driven execution revolves around the idea of applying transformation operations to a flow or stream of data when it is available.

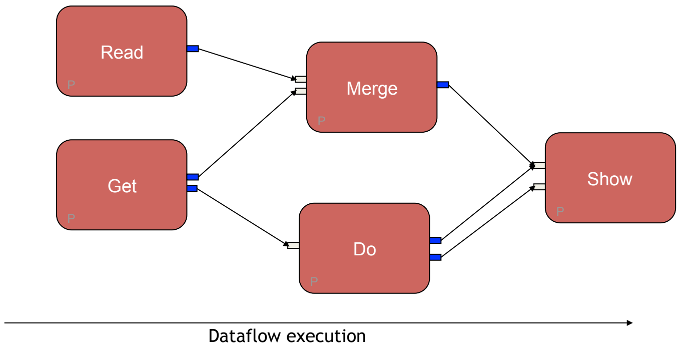
Meandre: The Dataflow Component

- Data dictates component execution semantics



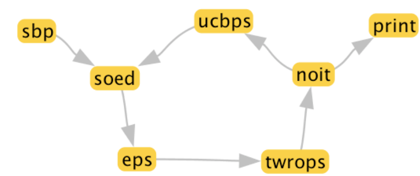
Meandre: Flow (Complex Tasks)

- A flow is a collection of connected components

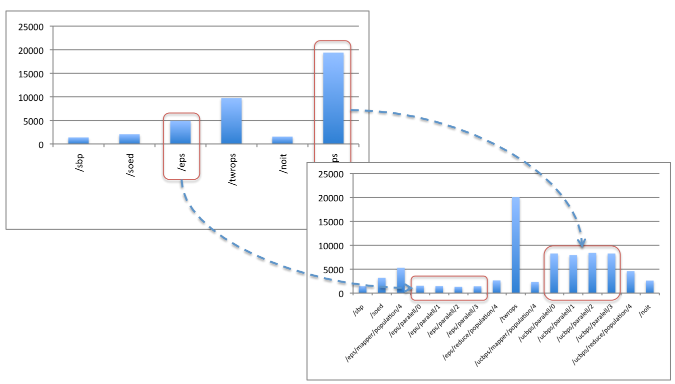


Your Point Being?

- Evolutionary algorithms can be modeled using data-intensive modeling
- Imagine a stream of individuals being process by components
- A single model implementation automatically parallelizable where needed



Collecting The Benefits



What does large scale mean?
Evolution as massive parallel processing
The challenges of data mining
Kaleidoscopic large scale data mining

Real-World Examples

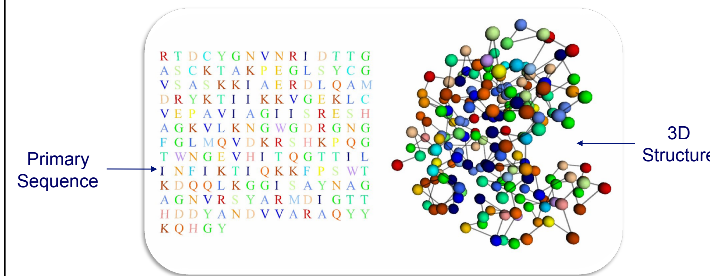
Wrapping up

Real-World Examples

- Example to present
  - Protein Structure & Contact Map Prediction (Bacardit et al., 2009)
  - Cancer prediction (Llora et al. 2007; Llora et al. 2009)
- A set of LCS applications to Data Mining is collected in (Bull et al., 2008)

Protein Structure Prediction

- Protein Structure Prediction (PSP) aims to predict the 3D structure of a protein based on its primary sequence

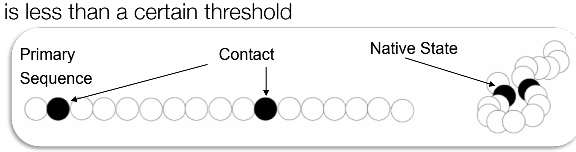


Protein Structure Prediction

- Beside the overall 3D PSP (an optimization problem), several structural aspects can be predicted for each protein residue
  - Coordination number
  - Solvent accessibility
  - Etc.
- These problems can be modelled in may ways:
  - Regression or classification problems
  - Low/high number of classes
  - Balanced/unbalanced classes
  - Adjustable number of attributes
- Ideal benchmarks !!
- http://www.infobiotic.net/PSPbenchmarks/

Contact Map Prediction

- Two residues of a chain are said to be in contact if their distance is less than a certain threshold

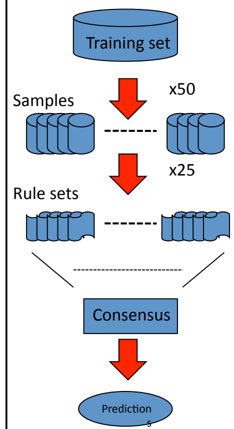


- Contact Map (CM): binary matrix that contains a 1 for a cell if the residues at the row & column are in contact, 0 otherwise
- This matrix is very sparse, in real proteins there are less than 2% of contacts
- Highly unbalanced dataset

Contact Map Prediction

- (Bacardit et al. 2009) participated in the CASP8 competition
- CASP = Critical Assessment of Techniques for Protein Structure Prediction. Biannual competition
- Every day, for about three months, the organizers release some protein sequences for which *nobody* knows the structure (129 sequences were released in CASP9, in 2010)
- Each prediction group is given three weeks to return their predictions
- If the machinery is not well oiled, it is not feasible to participate !!
- For CM, prediction groups have to return a list of predicted contacts (they are not interested in non-contacts) and, for each predicted pair of contacting residues, *a confidence level*
- The evaluation for CM ranks this list by the confidence, and calculates the accuracy of the top L/x predictions (L = length of chain, x = typically 10)

Contact Map Prediction: Hands on



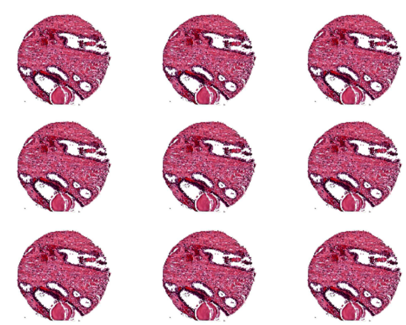
- Training set of 2413 proteins selected to represent a broad set of sequences
  - 32 million pairs of amino-acids (instances in the training set) with less than 2% of real contacts
  - Each instance is characterized by up to 631 attributes
- 50 samples of ~660000 examples are generated from the training set. Each sample contains two no-contact instances for each contact instance
- The BioHEL GBML method (Bacardit et al., 2009) was run 25 times on each sample
- An ensemble of 1250 rule sets (50 samples x 25 seeds) performs the contact maps predictions using simple consensus voting
- Confidence is computed based on the votes distribution in the ensemble

Results of Contact Map prediction

- The subset of the most difficult target (*Free Modelling targets)* of CASP9 were used to evaluate CM
- Out predictor obtained an average accuracy of 23.6%
- Do you think it is low?
  - It is more than 11 times higher than a random prediction
  - The predictor was the best *Ab Initio* method in the competition
- Overall, tackling this problem has forced us to address a broad range of bottlenecks in DM methods
  - Code bottlenecks
  - Memory footprint bottlenecks
  - Scalability bottlenecks

Prostate Cancer Diagnosis

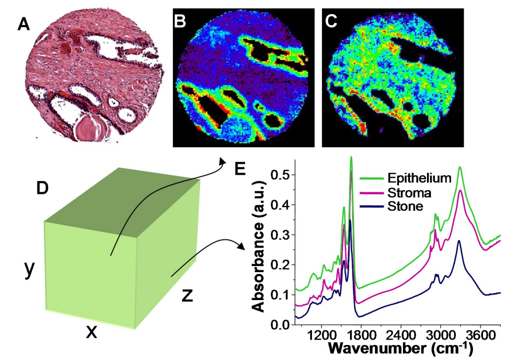
- Biopsy-staining-microscopy-manual recognition is <u>the</u> diagnosis procedure for the last 150 years.



Advances on Fourier Transform IR Imaging

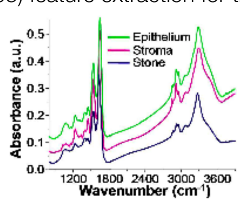
- Infrared spectroscopy is a classical technique for measuring chemical composition of specimens.
- At specific frequencies, the vibrational modes of molecules are resonant with the frequency of infrared light.
- Microscope has develop to the point that resolution that match a pixel with a cell (and keep improving).
- It allows to start from the same data (stained tissue)
- Generates larges volumes of data

Advances on Fourier Transform IR Imaging



Spectrum Analysis

- Microscope generate a lot of data
- Per spot the spectra signature requires GBs of storage
- Bhargava et al. (2005) feature extraction for tissue identification



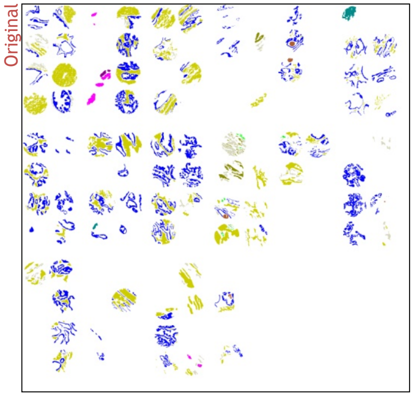
- More than 200 potential features per spectrum (cell/pixel)
- Firsts methodology that allowed tissue identification

Prostate Cancer Data

1. Tissue identification
   - Modeled as a supervised learning problem
   - (Features, tissue type)
   - The goal: Accurately retrieve epithelial tissue
2. Tissue diagnosis
   - Modeled as a supervised learning problem
   - (Features, diagnosis)
   - The goal: Accurately diagnose each cell (pixel) and aggregate those diagnosis to generate a spot (patient) diagnosis

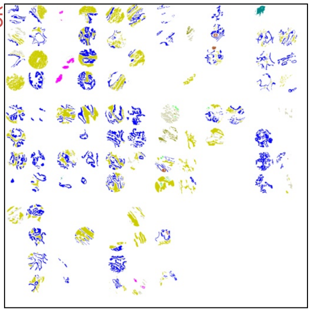
GBML Identifies Tissue Types

Original



GBML Identifies Tissue Types

OK

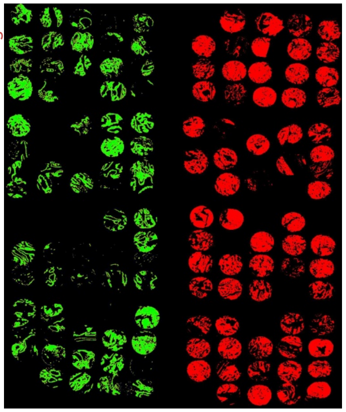


Misclassified

- Accuracy >96%
- Mistakes on minority classes (not targeted) and boundaries

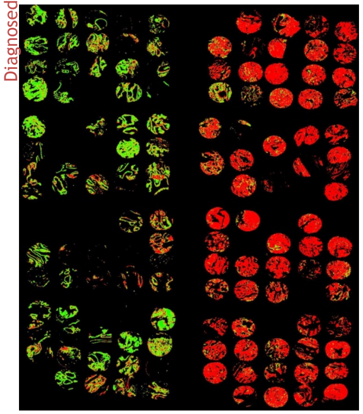
Filtered Tissue is Accurately Diagnosed

Original



Filtered Tissue is Accurately Diagnosed

Diagnosed



Filtered Tissue is Accurately Diagnosed

- Pixel crossvalidation accuracy using NAX (87.34%)
- Spot accuracy
  - 68 of 69 malignant spots
  - 70 of 71 benign spots
- Human-competitive computer-aided diagnosis system is possible (Humies 2007 Bronze award)
- First published results that fall in the range of human error (<5%)

Wrapping Up

Wrapping Up

- We have shown in this tutorial how GBML methods have high potential for mining large-scale datasets
- They are natural parallel processing machines
- Recent improvements in many dimensions of the learning process
  - Representations
  - Learning paradigms
  - Inference mechanisms
  - Hybridization

The Game Has a New Name

- The exception is becoming norm
  - Efficient parallel designs
  - Efficiency enhancement methods
  - Hardware support (SSE, CUDA, etc.)
- However, all these components cannot be used blindly, they have to be adjusted properly, accordingly to the characteristics/dimensions of the problem

Better Understanding

- Theoretical analysis of the different facets of a GBML system can help
- Understand better why/when can the components perform well
- Design robust policies that can take the best of the techniques at hand
- Provide insight on parameterization of methods
  - If we would like the community to use GBML methods, we have to make them easy to use
- Some work already exists (Butz, 2006; Franco et al., 2011), but we still have a long road ahead of us

Do not Be Shy

- GBML systems are highly flexible, with good explanatory power, and can have good scalability
- Go and give it a shoot!

References

- http://www.ncbi.nlm.nih.gov/Genbank/index.html
- http://www.netflixprize.com/
- V. Reinke, Germline genomics (January 20, 2006), WormBook, ed. The C. elegans Research Community, WormBook, doi/10.1895/wormbook.1.74.1, http://www.wormbook.org
- Bernadó, E., Ho, T.K., Domain of Competence of XCS Classifier System in Complexity Measurement Space, IEEE Transactions on Evolutionary Computation, 9: 82-104, 2005.
- "Physicists brace themselves for lhc 'data avalanche'." www.nature.com/news/2008/080722/full/news.2008.967.html
- M. Pop and S. L. Salzberg, "Bioinformatics challenges of new sequencing technology," Trends in Genetics, vol. 24, no. 3, pp. 142 – 149, 2008
- http://www.hdfgroup.org/HDF5
- K. Sastry, "Principled Efficiency-Enhancement Techniques", GECCO-2005 Tutorial
- A.A. Freitas, "Data Mining and Knowledge Discovery with Evolutionary Algorithms", Springer-Verlag, 2002
- J. Bacardit, Pittsburgh Genetics-Based Machine Learning in the Data Mining era: Representations, generalization, and run-time. PhD thesis, Ramon Llull University, Barcelona, Spain, 2004

References

- Jaume Bacardit, David E. Goldberg, Martin V. Butz, Xavier Llorà and Josep M. Garrell, Speeding-up Pittsburgh Learning Classifier Systems: Modeling Time and Accuracy, 8th International Conference on Parallel Problem Solving from Nature - PPSN VIII
- D. Song, M.I. Heywood and A.N. Zincir-Heywood, Training genetic programming on half a million patterns: an example from anomaly detection, IEEE Transactions on Evolutionary Computation, vol. 9, no. 3, pp 225-239, 2005
- Llora, X., Priya, A., and Bhragava, R. (2007), Observer-Invariant Histopathology using Genetics-Based Machine Learning. Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2007) , 2098–2105
- Giráldez R, Aguilar-Ruiz JS, Santos JCR (2005) Knowledge-based fast evaluation for evolutionary learning. IEEE Transactions on Systems, Man, and Cybernetics, Part C 35 (2):254–261
- J. Bacardit, E. K. Burke, and N. Krasnogor. Improving the scalability of rule-based evolutionary learning. Memetic Computing, 1(1):55-67, 2009
- M. V. Butz, P. L. Lanzi, X. Llorà, and D. Loiacono. An analysis of matching in learning classifier systems.In GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation, pp. 1349–1356. ACM, 2008.
- Llorà, X., Sastry, K., Yu, T., and Goldberg, D. E. Do not match, inherit: fitness surrogates for genetics-based machine learning techniques. In Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation, pp 1798-1805, ACM, 2007

References

- Orriols-Puig, A., Bernadó-Mansilla, E., Sastry, K., and Goldberg, D. E. Substructrual surrogates for learning decomposable classification problems: implementation and first results. 10th International Workshop on Learning Classifier Systems, 2007
- J. Bacardit and N. Krasnogor, Performance and Efficiency of Memetic Pittsburgh Learning Classifier Systems, Evolutionary Computation Journal, 17(3):307-342, 2009
- G. Wilson and W. Banzhaf, "Linear genetic programming gpgpu on microsoft's xbox 360," in Proceedings of the 2008 Congress on Evolutionary Computation, pp. 378-385. IEEE Press, 2008
- http://www.gpgpgpu.com/
- J. Bacardit and N. Krasnogor. "Empirical evaluation of ensemble techniques for a Pittsburgh Learning Classifier System". Learning Classifier Systems. LNAI 4998, pp. 255-268, 2008, Springer
- http://www.infobiotic.net/PSPbenchmarks/
- J. Bacardit, M. Stout, J.D. Hirst, K. Sastry, X. Llorà and N. Krasnogor. Automated Alphabet Reduction Method with Evolutionary Algorithms for Protein Structure Prediction In Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO2007), pp. 346-353, ACM Press, 2007
- Goldberg, D. E., Sastry, K. and Llora, X. (2007), Toward routine billion-variable optimization using genetic algorithms. Complexity , 12(3), 27–29.

References

- G. Venturini. SIA: A supervised inductive algorithm with genetic search for learning attributesbased concepts. In: Brazdil PB (ed) Machine Learning: ECML-93 - Proc. of theEuropean Conference on Machine Learning, Springer-Verlag, Berlin, Heidelberg, pp 280–296, 1993
- J. Rissanen J. Modeling by shortest data description. Automatica vol. 14:465–471, 1978
- L. Bull, E. Bernadó-Mansilla and J. Holmes (editors), Learning Classifier Systems in Data Mining. Springer, 2008
- Alba, E., Ed. Parallel Metaheuristics. Wiley, 2007.
- Cantu-Paz, E. Efficient and Accurate Parallel Genetic Algorithms. Springer, 2000.
- Llora, X. E2K: evolution to knowledge. SIGEVOlution 1, 3 (2006), 10–17.
- Llora, X. Genetic Based Machine Learning using Fine-grained Parallelism for Data Mining. PhD thesis, Enginyeria i Arquitectura La Salle. Ramon Llull University, Barcelona, February, 2002.RFC2413, The Dublin Core Metadata Initiative, 2008.
- Llora, X., Acs, B., Auvil, L., Capitanu, B., Welge, M., and Goldberg, D. E. Meandre: Semantic-driven data-intensive flows in the clouds. In Proceedings of the 4th IEEE International Conference on e-Science (2008), IEEE press, pp. 238–245.
- M. Butz, Rule-Based Evolutionary Online Learning Systems: A Principled Approach to LCS Analysis and Design, Studies in Fuzziness and Soft Computing, vol 109. Springe,r, 2006

References

- Hadoop (http://hadoop.apache.org/core/)
- Meandre (http://seasr.org/meandre)
- Dean, J. & Ghemawat, S. MapReduce: Simplified Data Processing in Large Clusters, OSDI'04: Sixth Symposium on Operating System Design and Implementation, San Francisco, CA, December, 2004.
- Franco, M. A., Krasnogor, N., and Bacardit, J. Speeding up the evaluation of evolutionary learning systems using GPGPUs. In *Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation* - GECCO '10. pp 1039-1046, 2010
- A. Verma, X. Llora, S. Venkataram, D.E. Goldberg and R. Campbell, *Scaling eCGA Model Building via Data Intensive Computing, In Proceedings of WCCI2010, IEEE press, pp 1-8, 2010*
- J. Bacardit, M. Stout, J.D. Hirst, A. Valencia, R.E. Smith and N. Krasnogor. Automated Alphabet Reduction for Protein Datasets. BMC Bioinformatics 10:6, 2009
- M. Franco, N. Krasnogor and J. Bacardit. Modelling the Initialisation Stage of the ALKR representation for Discrete Domains and GABIL encoding, In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation* - GECCO '11, to appear.

Large Scale Data Mining using Genetics-Based Machine Learning

Jaume Bacardit

School of Computer Science &
School of Bioscience
University Nottingham
Nottingham, UK

jqb@cs.nott.ac.uk

Xavier Llorà

Data-Intensive Technologies and Applications
National Center for Supercomputing Applications
University of Illinois at Urbana-Champaign
Urbana, IL 61801

xllora@illinois.edu

GECCO 2011 Tutorial, July 12th 2011