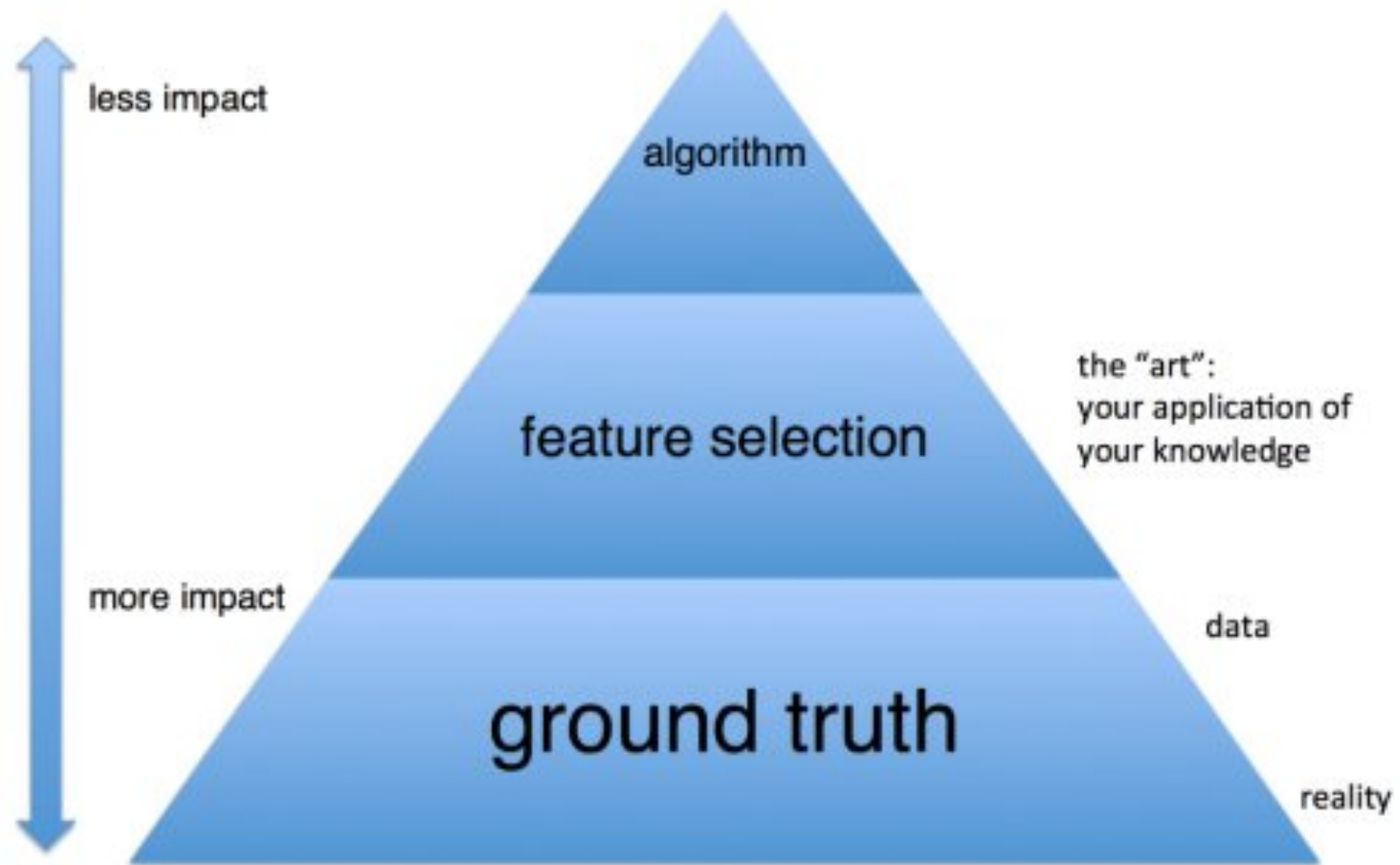


Machine Learning

From systems engineering to biomedical applications

Alcides Fonseca
alcides@ciencias.ulisboa.pt





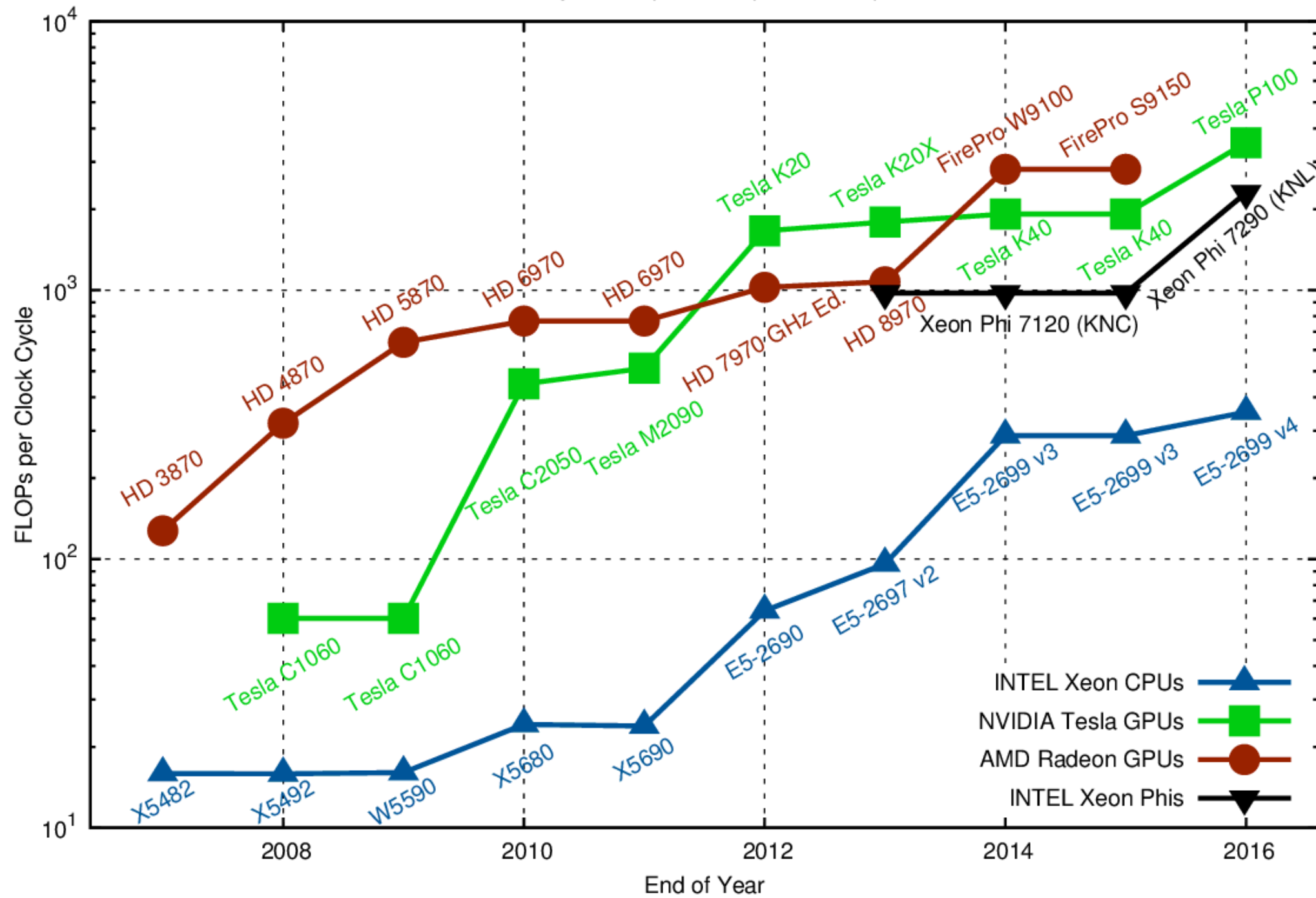
```
from sklearn import tree

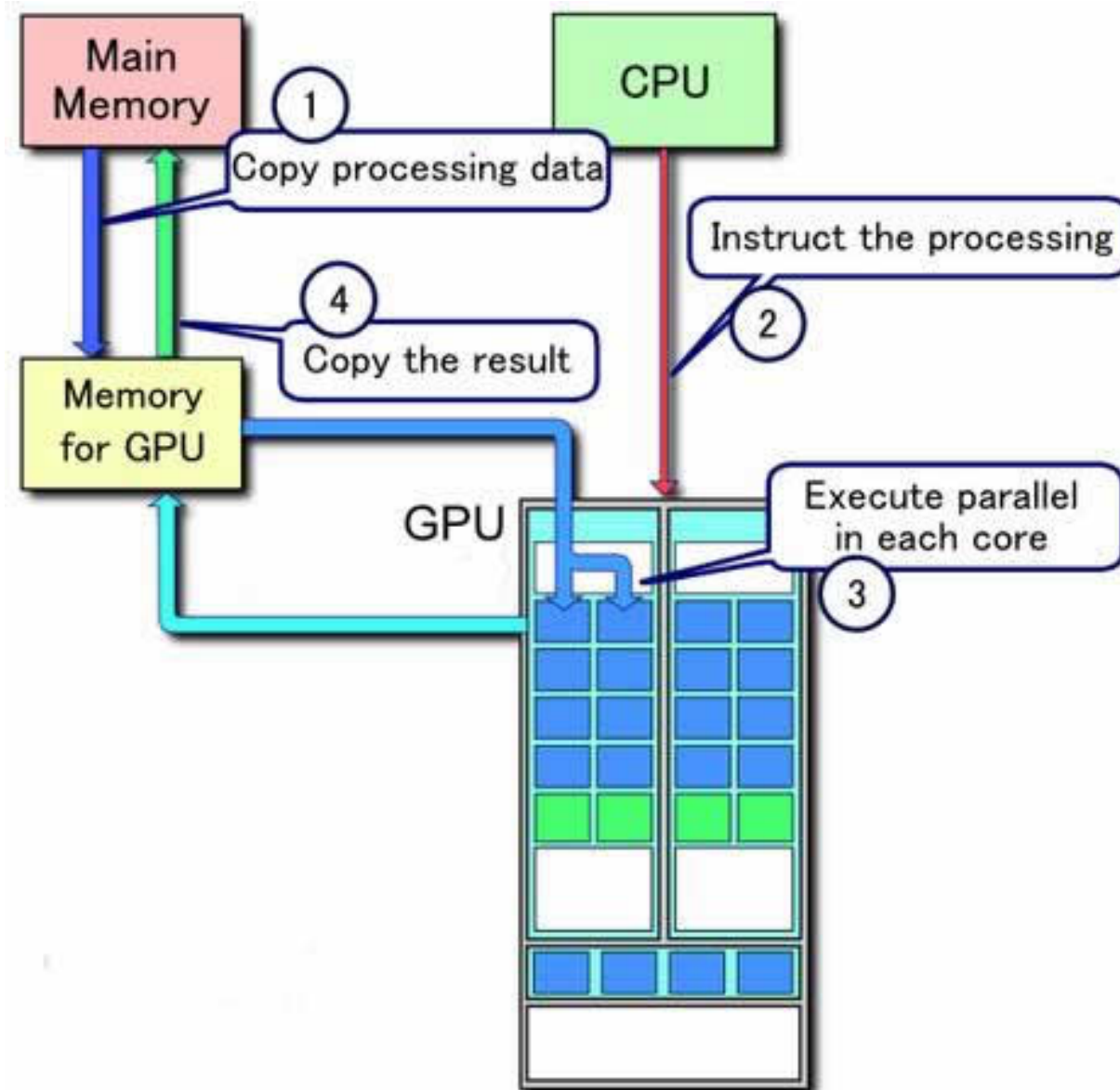
c = tree.DecisionTreeClassifier()
c.fit(X, y)
y_new = c.predict(X_new)
```

	Terror	Comedy	#views	#views_from_people_who_like_comedy	Liked
Jurassic World	0.3	0.5	8345	1431	1
Walking Dead	1	0	20	1	1
SEAL Team	0	0	10	4	0
			X		y
Get Shorty	0	0.7	4	2	?
			X_new		y_new

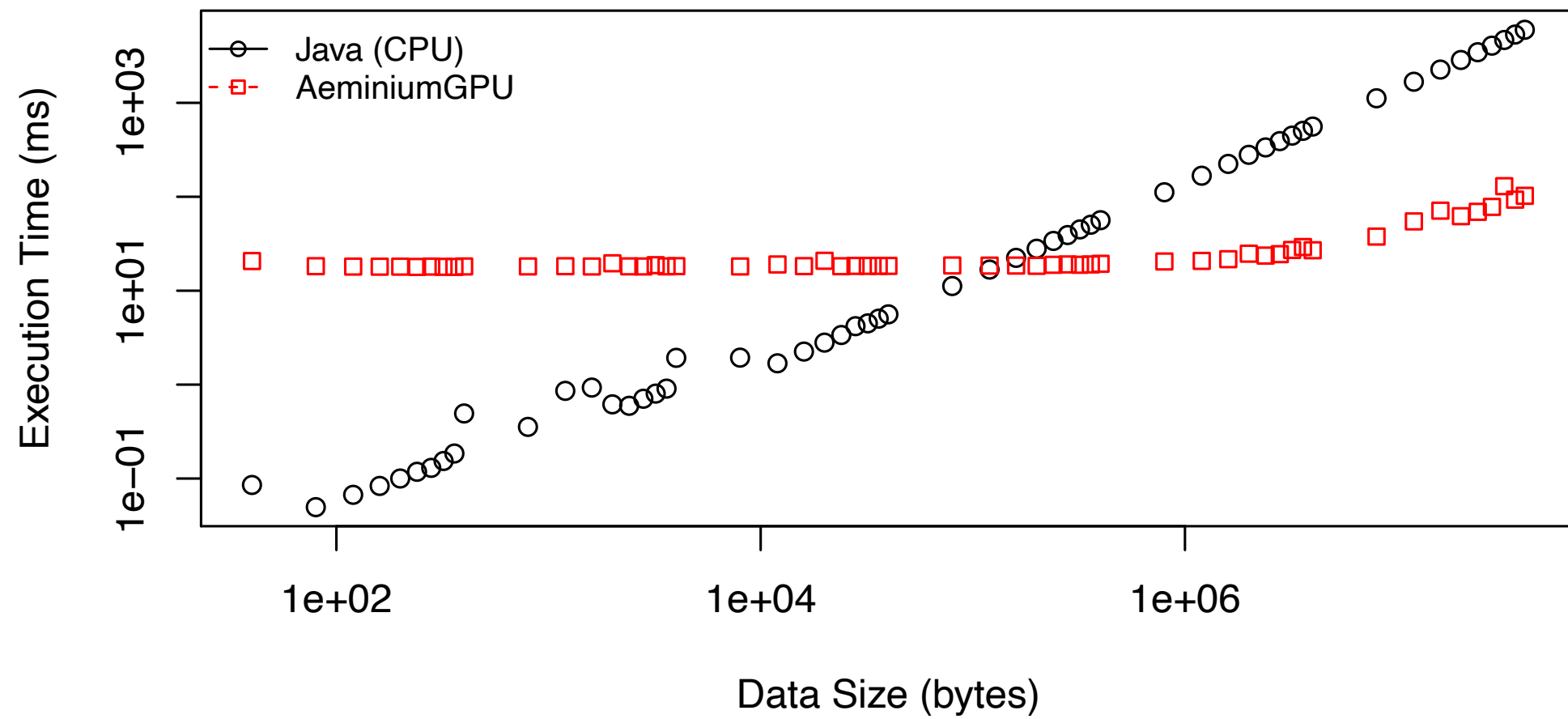
CPU vs GPU decision

Theoretical Peak Floating Point Operations per Clock Cycle, Double Precision

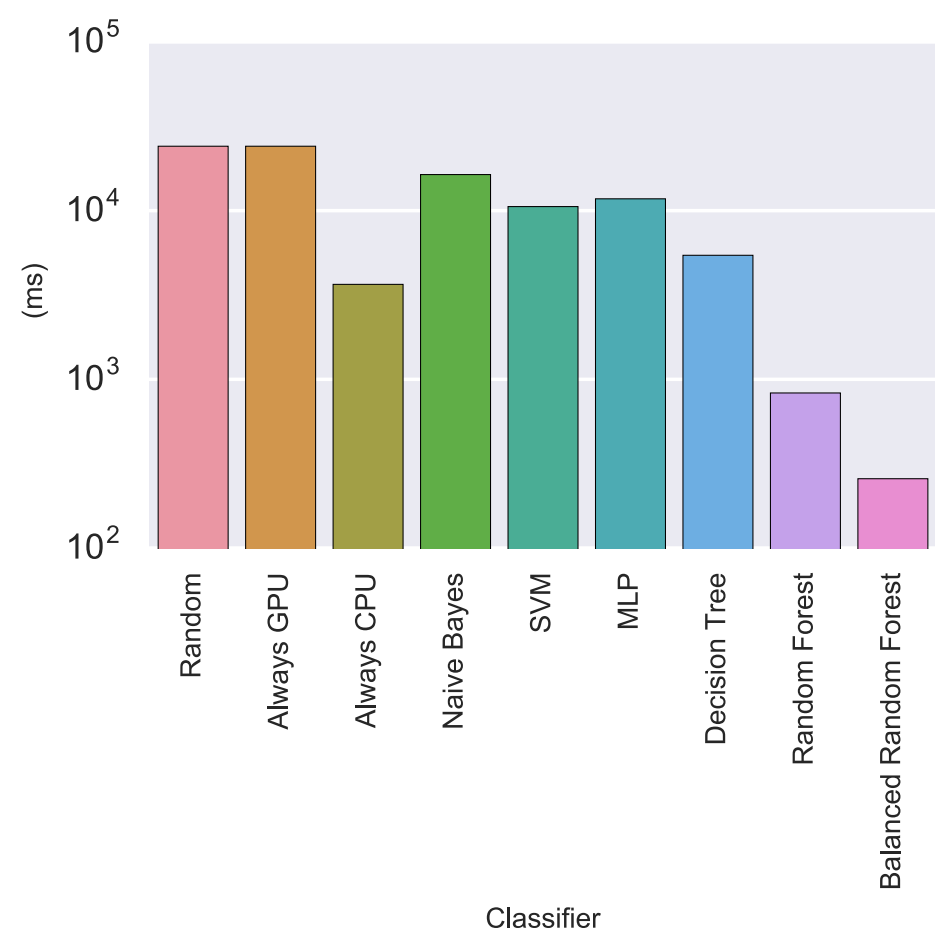
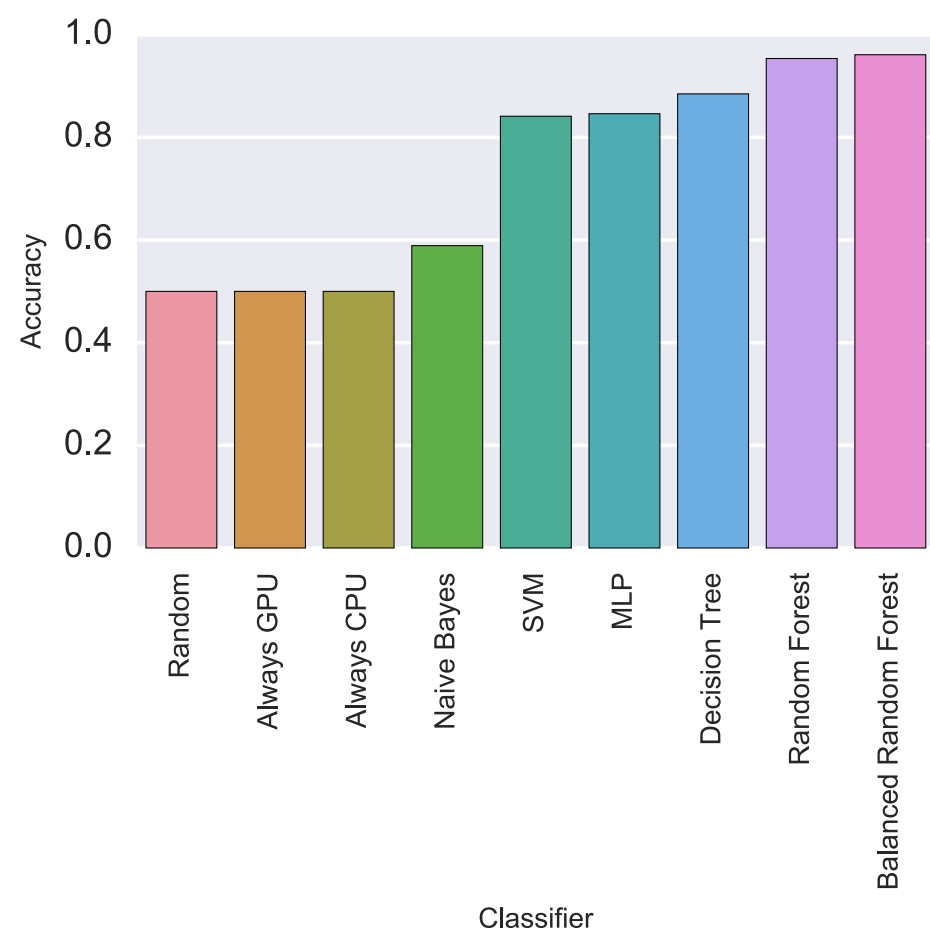




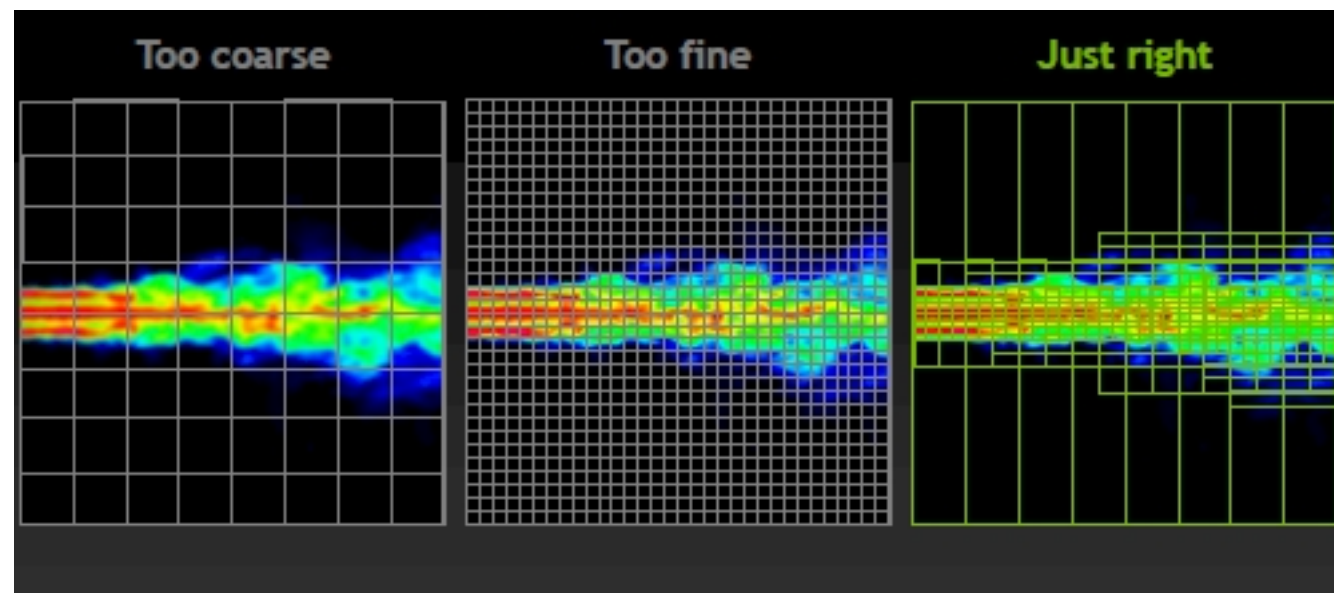
integral program – CPU vs GPU on AeminiumGPU

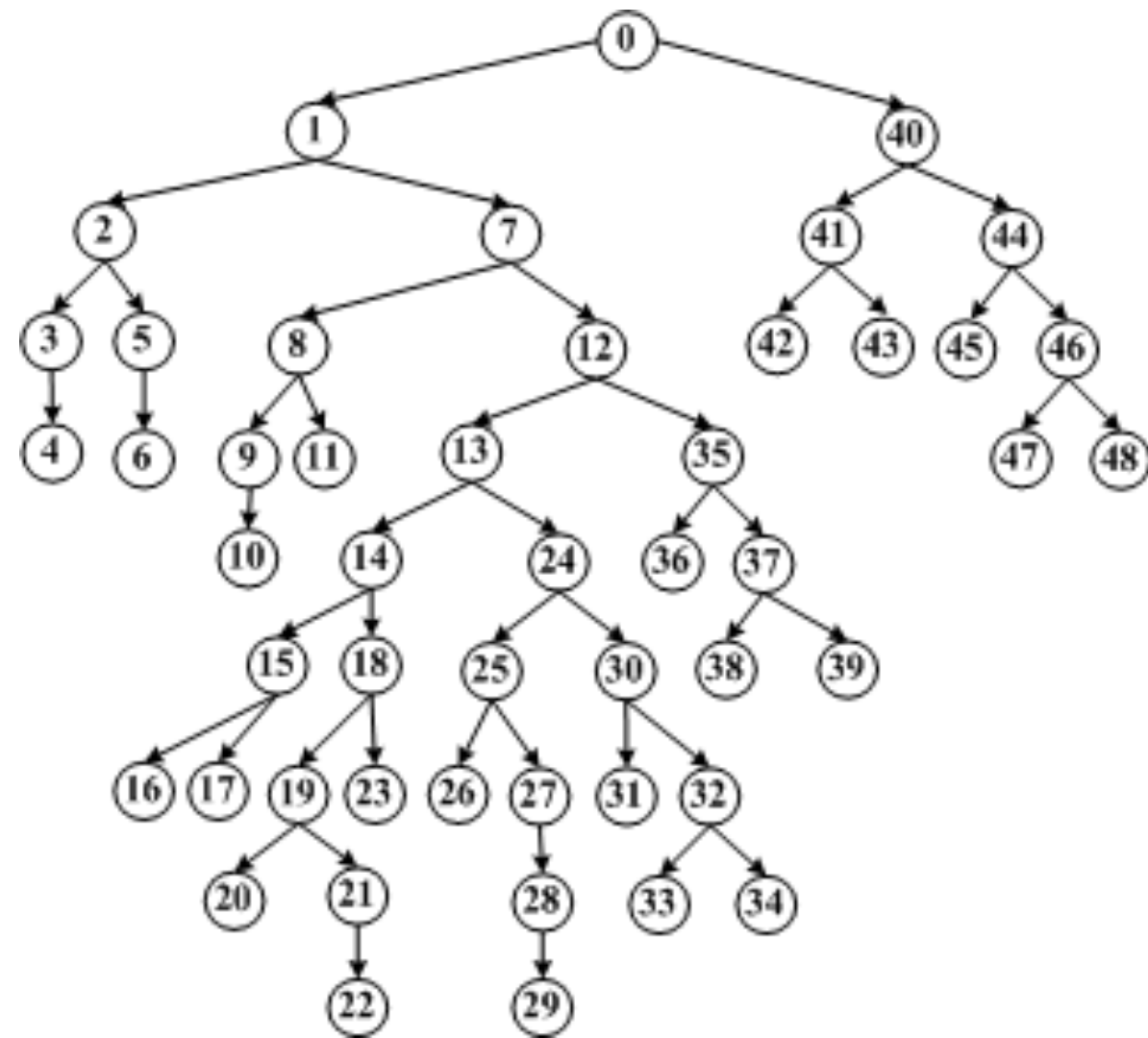


Name	Size	C/R	Description
OuterAccess	3	C	Global GPU memory read.
InnerAccess	3	C	Local (thread-group) memory read. This area of the memory is faster than the global one.
ConstantAccess	3	C	Constant (read-only) memory read. This memory is faster on some GPU models.
OuterWrite	3	C	Write in global memory.
InnerWrite	3	C	Write in local memory, which is also faster than in global.
BasicOps	3	C	Simplest and fastest instructions. Include arithmetic, logical and binary operators.
TrigFuns	3	C	Trigonometric functions, including <i>sin</i> , <i>cos</i> , <i>tan</i> , <i>asin</i> , <i>acos</i> and <i>atan</i> .
PowFuns	3	C	<i>pow</i> , <i>log</i> and <i>sqrt</i> functions
CmpFuns	3	C	<i>max</i> and <i>min</i> functions
Branches	3	C	Number of possible branching instructions such as <i>for</i> , <i>if</i> and <i>whiles</i>
DataTo	1	R	Size of input data transferred to the GPU in bytes.
DataFrom	1	R	Size of output data transferred from the GPU in bytes.
ProgType	1	R	One of the following values: Map, Reduce, PartialReduce or MapReduce, which are the different types of operations supported by ÆminiumGPU.

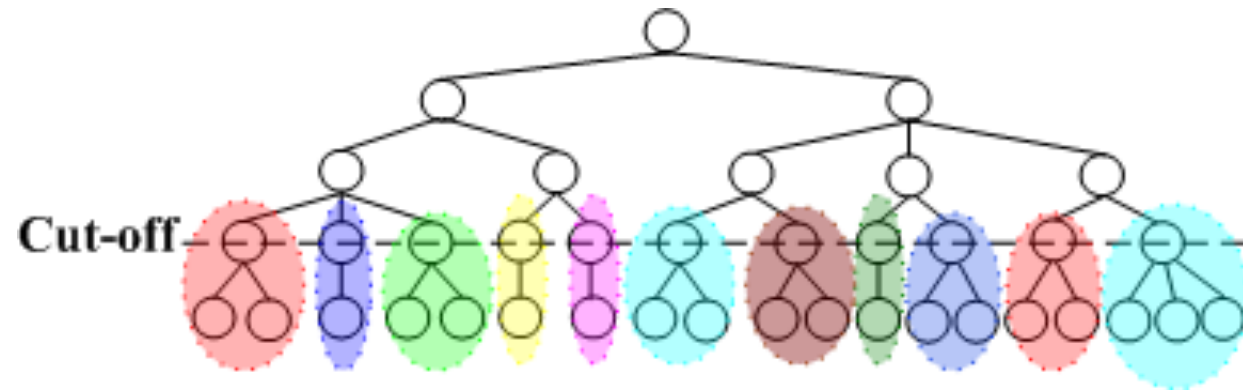


Runtime cut-off algorithm decision

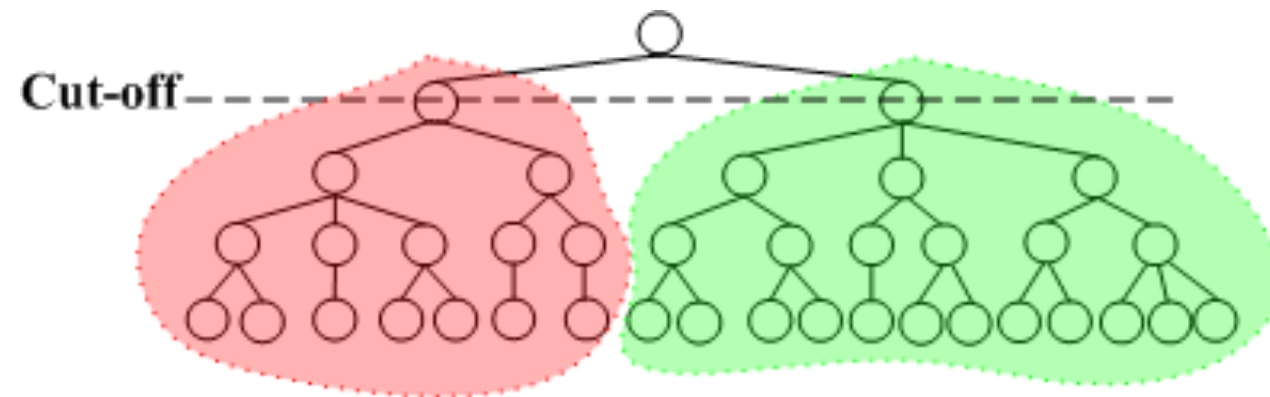




Cut-off = 3



Cut-off = 1



```
int f(int n) {  
    if (RuntimeManager.shouldSeq())  
        return jpar_sequential_version_of_f(n);  
  
    if (n < 2) {  
        return n;  
    }  
    Future<Integer> b_tmp = new Future<Integer>(task -> f(n-2));  
    Future<Integer> a_tmp = new Future<Integer>(task -> f(n-1));  
    int a = a_tmp.get();  
    int b = b_tmp.get();  
    return a + b;  
}
```

The No Free Lunch Theorem for Cut-off Algorithms

	lower bound	mean	upper bound
cut-off algorithm			
atc2l12	4.808	5.923	5.615
loadbased	4.808	4.962	5.308
maxlevel12	4.577	4.231	5.000
maxtasks2	4.038	4.154	4.077
maxtasks2ss16	4.654	4.154	4.154
maxtasksinq2	4.462	4.000	4.269
stacksize16	5.385	5.038	4.577
surplus3	4.500	4.269	4.231

Ranking of Cut-off Algorithms across different programs

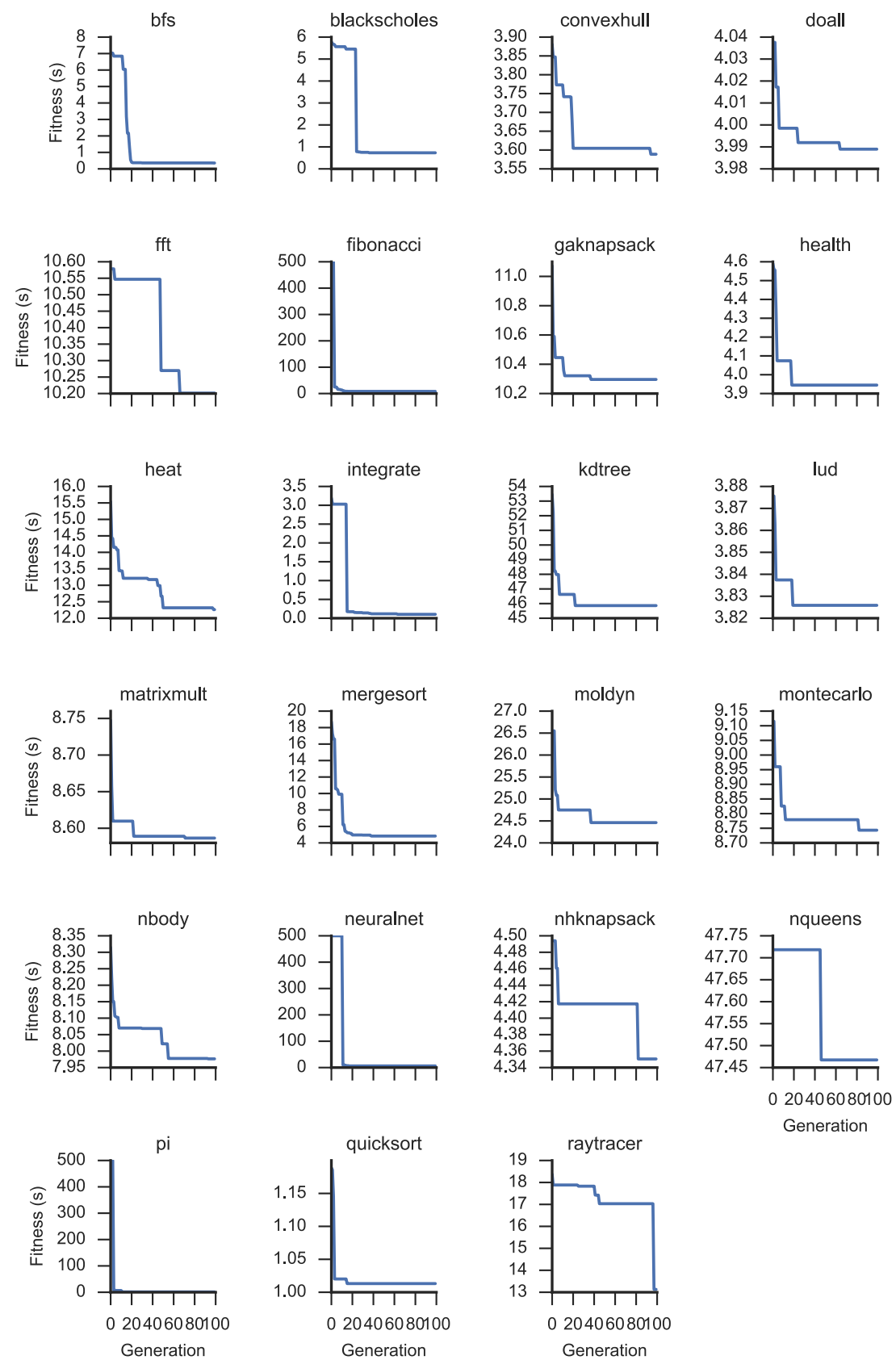
Genetic Algorithm

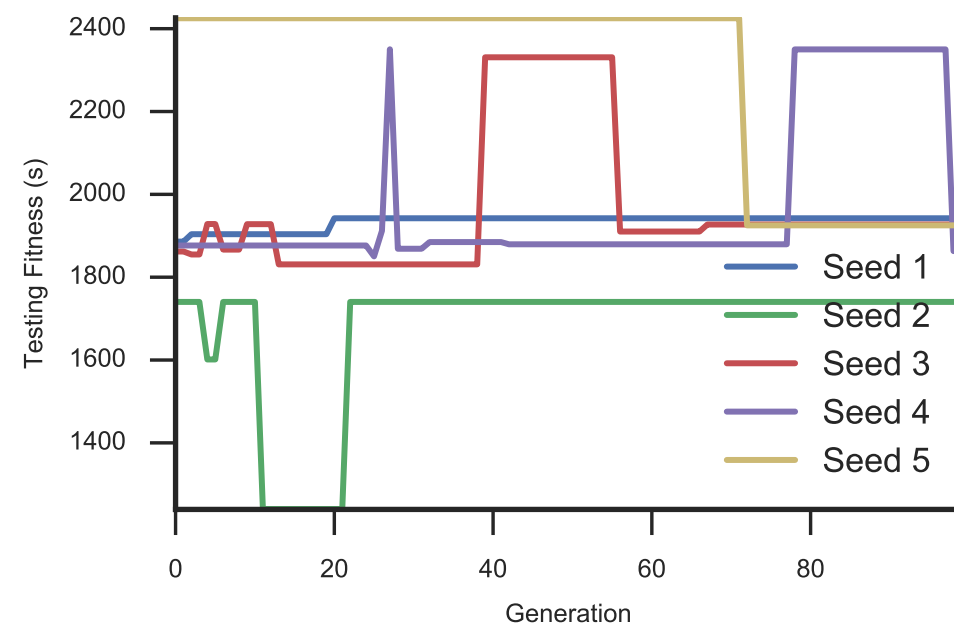
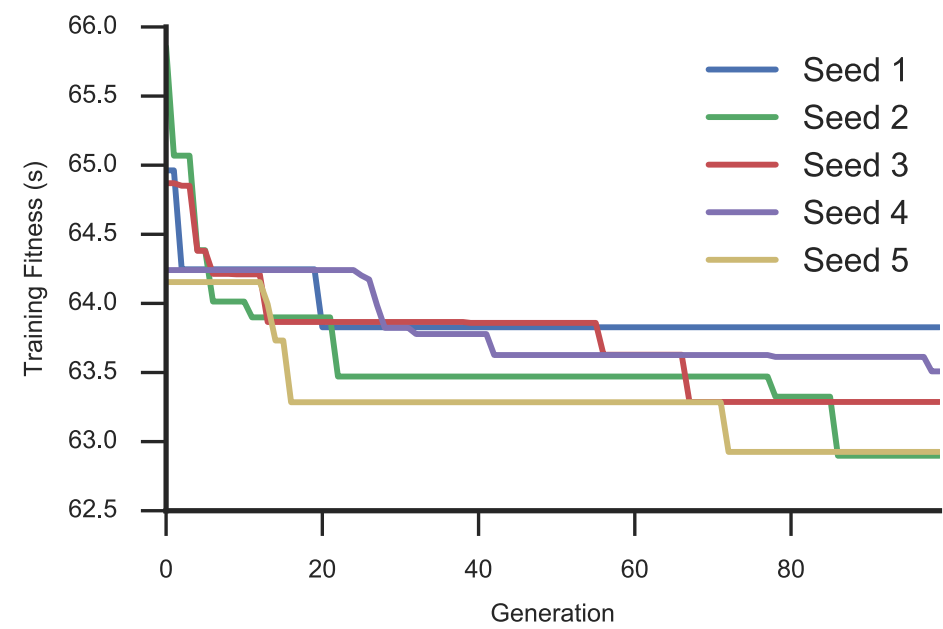
parallelize if: $\text{Var}_1 < \text{Threshold}_1 \uparrow_1 \text{Var}_2 < \text{Threshold}_2 \uparrow_2 \text{Var}_3 < \text{Threshold}_3$

Gene	Description
var_1	One of 11 options in Table 1
$threshold_1$	Integer between 0 and 100
$\uparrow_1$	and or or binary operators
var_2	One of 11 options in Table 1
$threshold_2$	Integer between 0 and 100
$\uparrow_2$	and or or binary operators
var_3	One of 11 options in Table 1
$threshold_3$	Integer between 0 and 100
$granularityCacheCalls$	Integer between 0 and 1000
$granularityCacheTime$	Integer between 0 and 1000
PPS	Integer between 0 and 100
$Stealing$	One of: StealFromMaxQueue, MinLevel, Revenge and SequentialReverseScan
$maxQueueSize$	Integer between 0 and 1000
$unparkingInterval$	Integer between 0 and 1000

Table 1. Possible Cut-off Variables

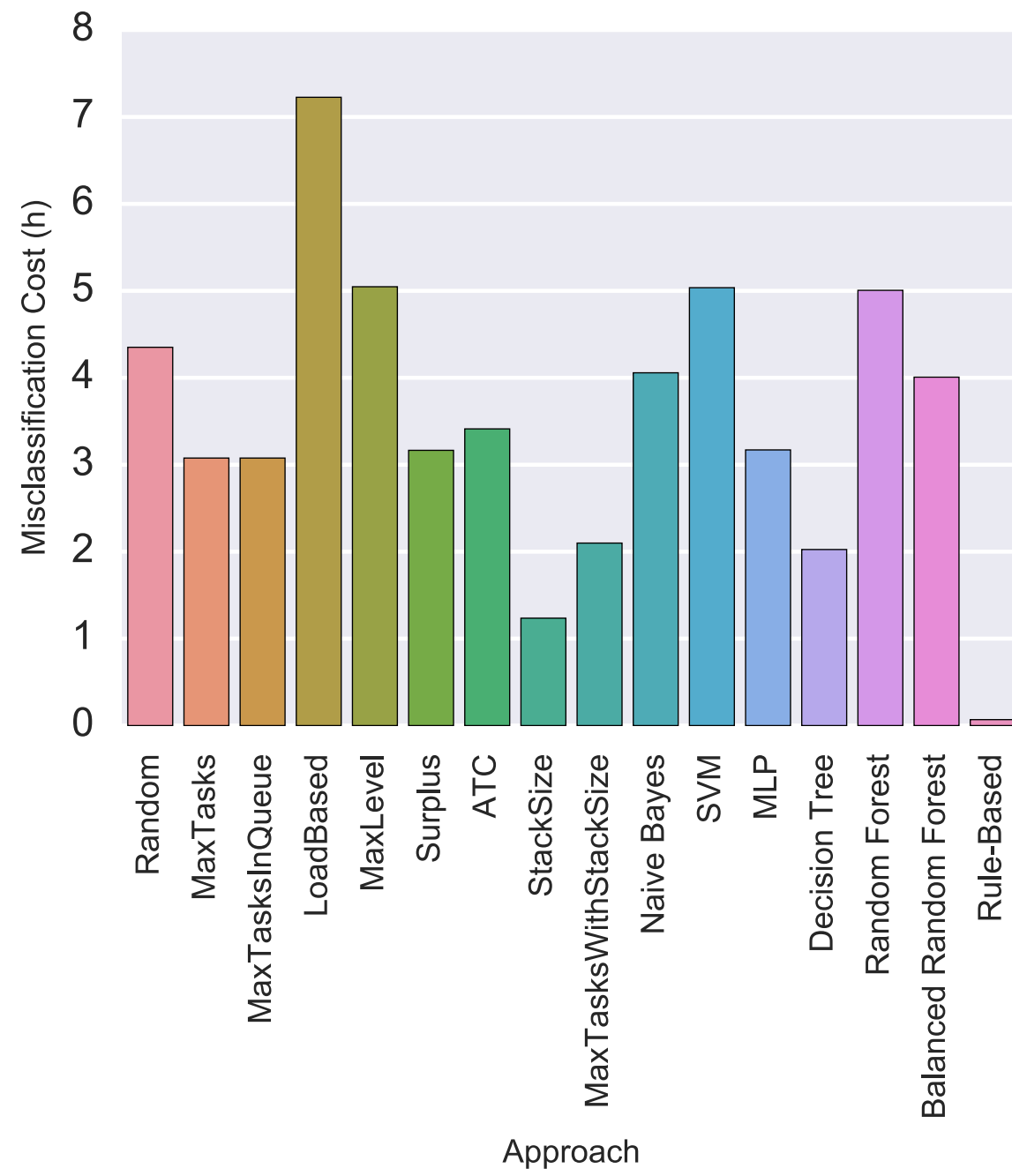
Variable	Description
queueSize	Length of the current queue
level	Depth of the parallel recursion
stacksize	Number of allocated stacks
activeThreads	Number of threads executing tasks
idleThreads	Number of threads not executing tasks
surplus	Length difference between local and average queues
totalTasks	Number of active and pending tasks
emptyQueues	Number of empty queues
memory	Percentage of used memory in the machine
cpuLoad	Percentage of CPU load in the machine
true	-1000, so the condition is always true.





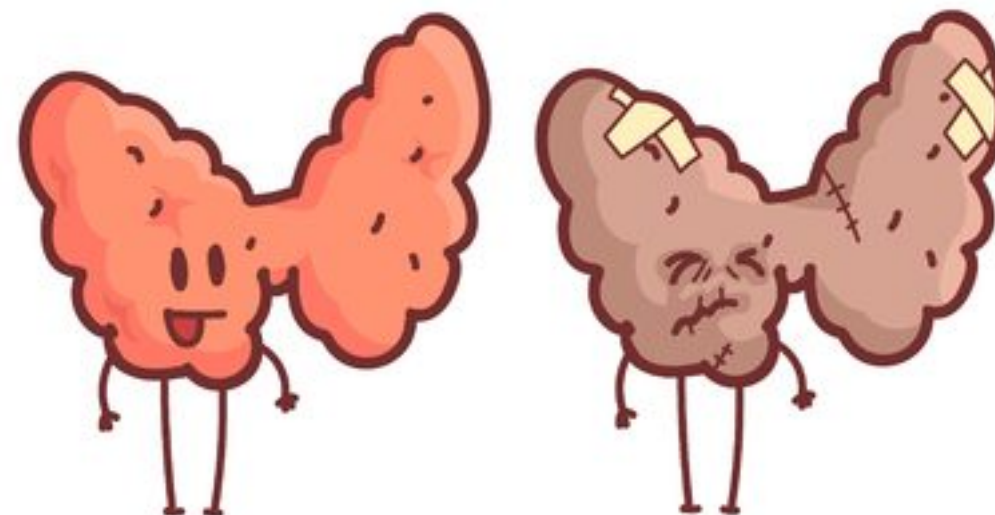
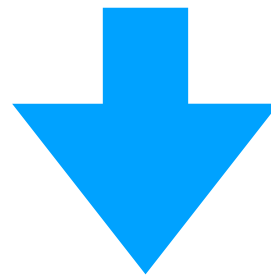
Program Features

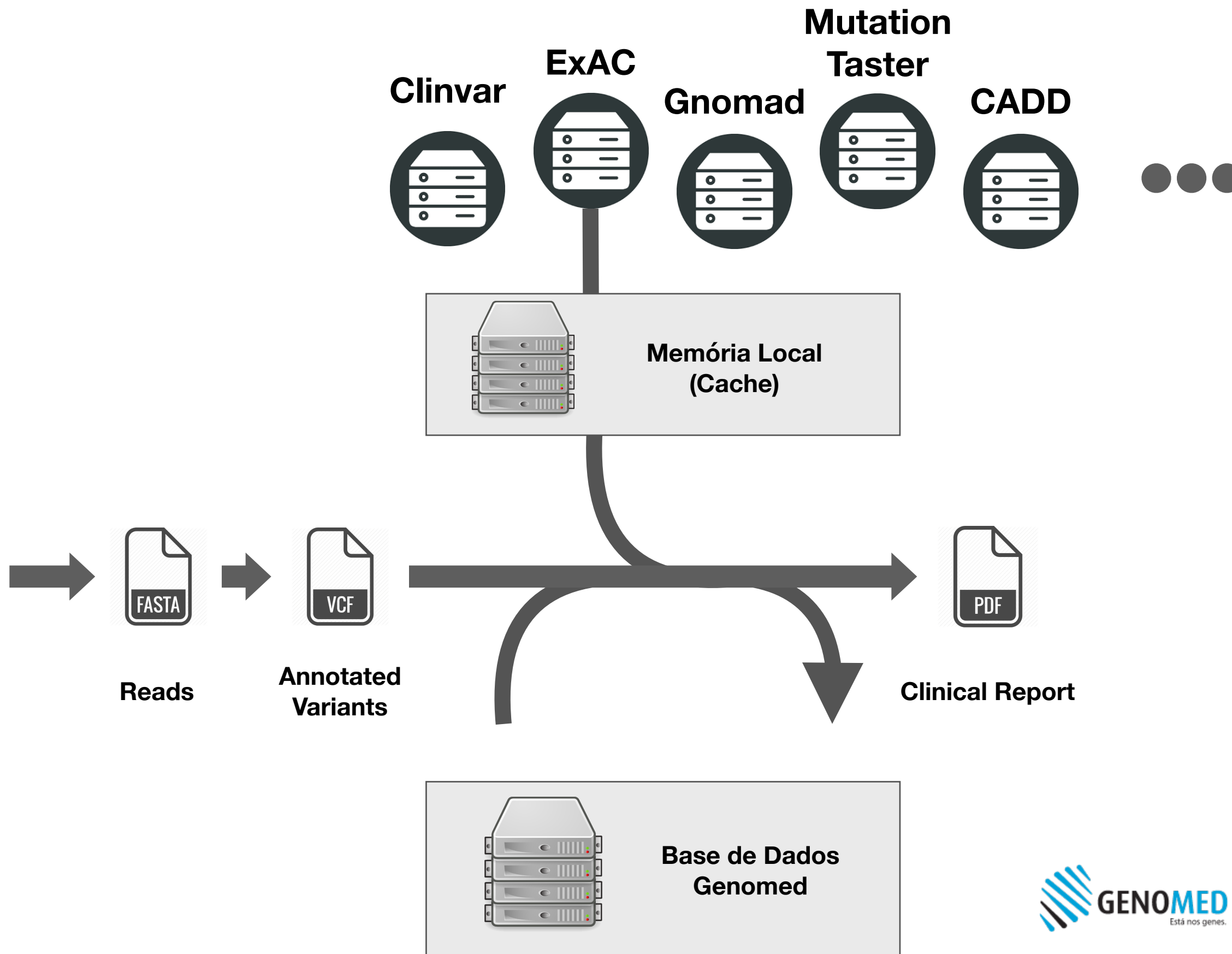
- - For-Loop (or not)
- - Unbalance
- - # Kernels
- - Branching Factor
- - Nesting Level

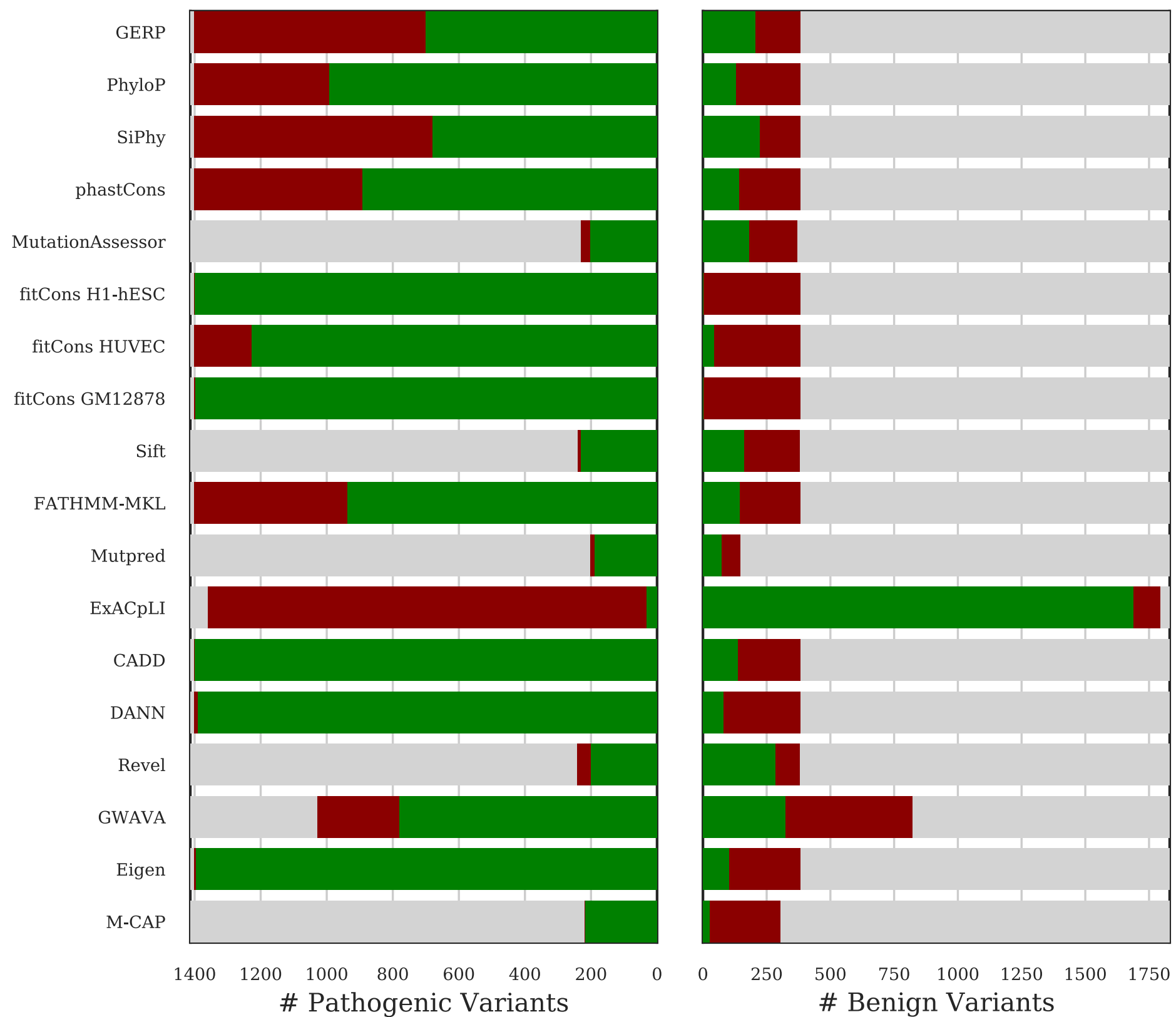


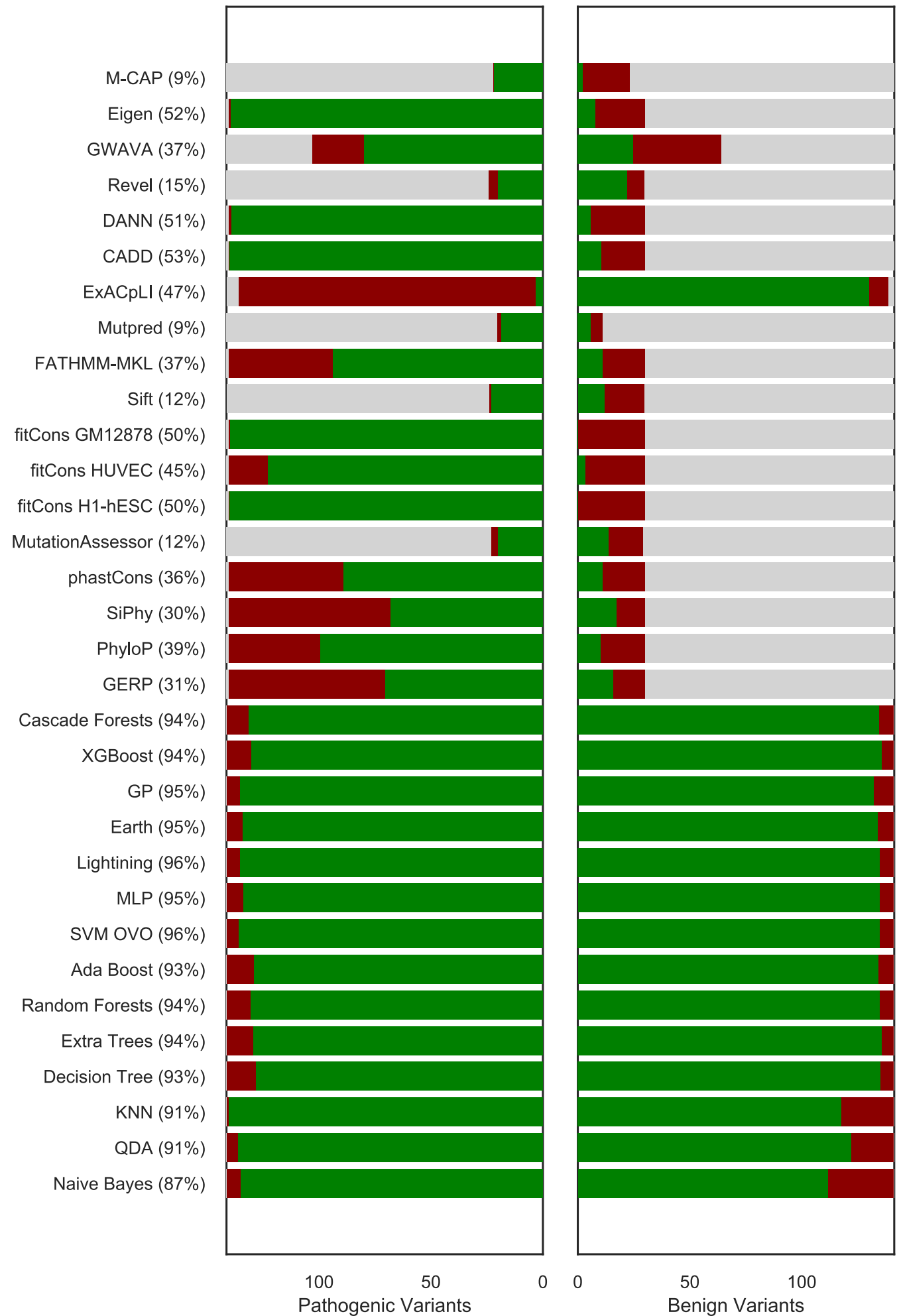
Gene Variant Pathogenicity Prediction

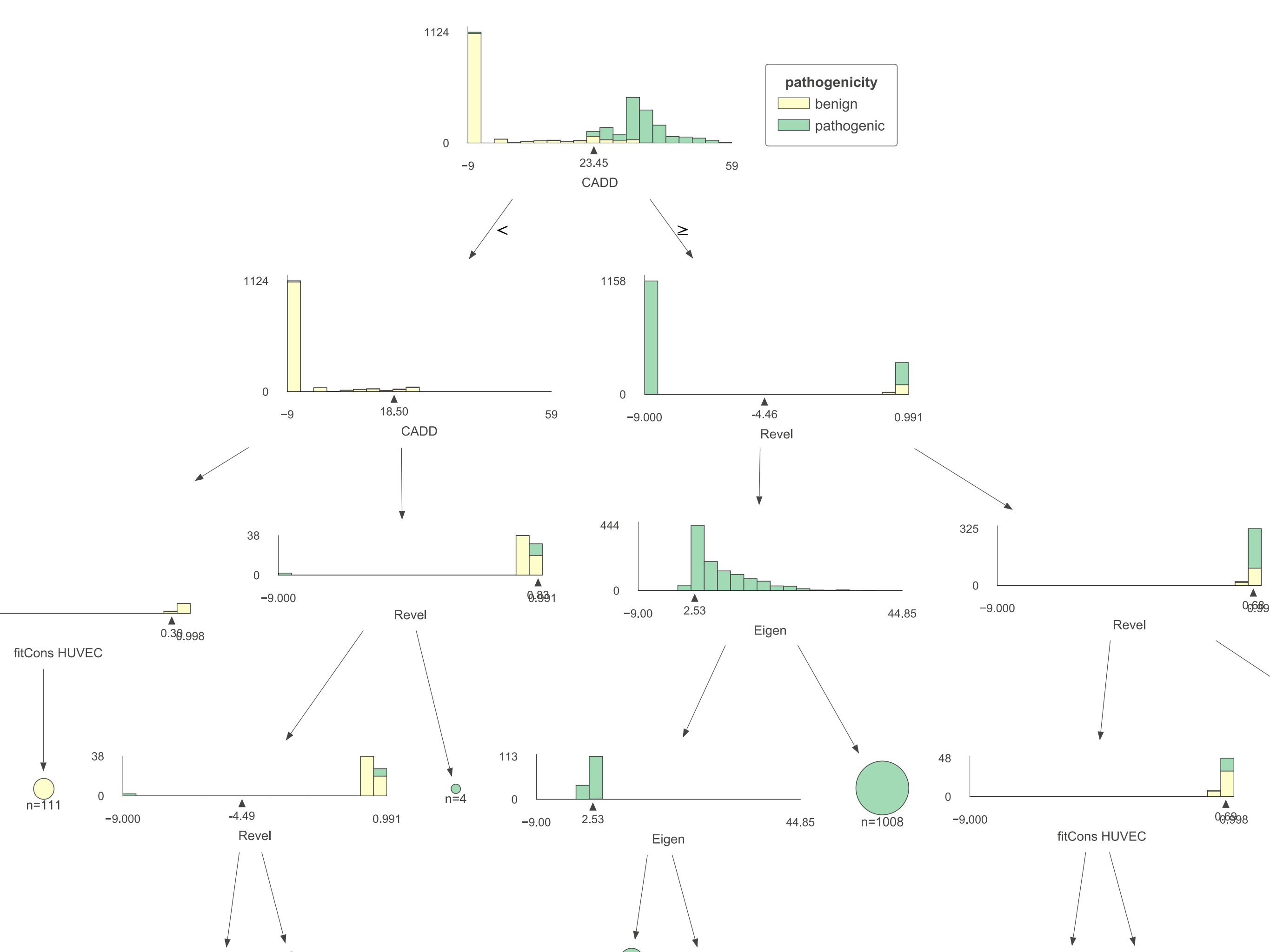
ACTGAGTTCCCTGGAACGGGACGCCATAG
TACTGAGTTCCCTGGAACGGGACGCCATAG
CCGTCTGGTAGGACACCCAGCCCCGTCTGA
TTCCGAGTTCCCTGGAACGGGACGCCATAG
CTTCCGAGTTCCCTGGAACGGGACGCCATAG
TCCGAGTTCCCTGGAACGGGACGCCATAG
GGATAACCGTGGTAATTCTAGAGCTAATA
ACGCCATAGAGGGTGAGAGCCCCGTCTGA
TTCCGAGTTCCCTGGAACGGGACGCCATAG
CGGGACGCCATAGAGGGTGAGAGCCCCGT
CGTCTGGTAGGACACCCAGCCCCGTCTGA











Machine Learning

From systems engineering to biomedical applications

Alcides Fonseca
alcides@ciencias.ulisboa.pt

