

University College London

Reducing Energy Consumption Using Genetic Improvement

Bobby R. Bruce
Justyna Petke
Mark Harman

How Many Wind Turbines do you need to Power the entire ICT infrastructure?

- ❖ Average 2MW wind turbine will generate 3.7GWh per year (20% capacity factor)



The Answer

- ❖ ICT consumption
- ❖ Therefore, 2
- ❖ This would land



2012 (4.7%)*
e required
kilometres of

*Lannoo, Bart, et al. "Overview of ICT energy consumption" (2013)

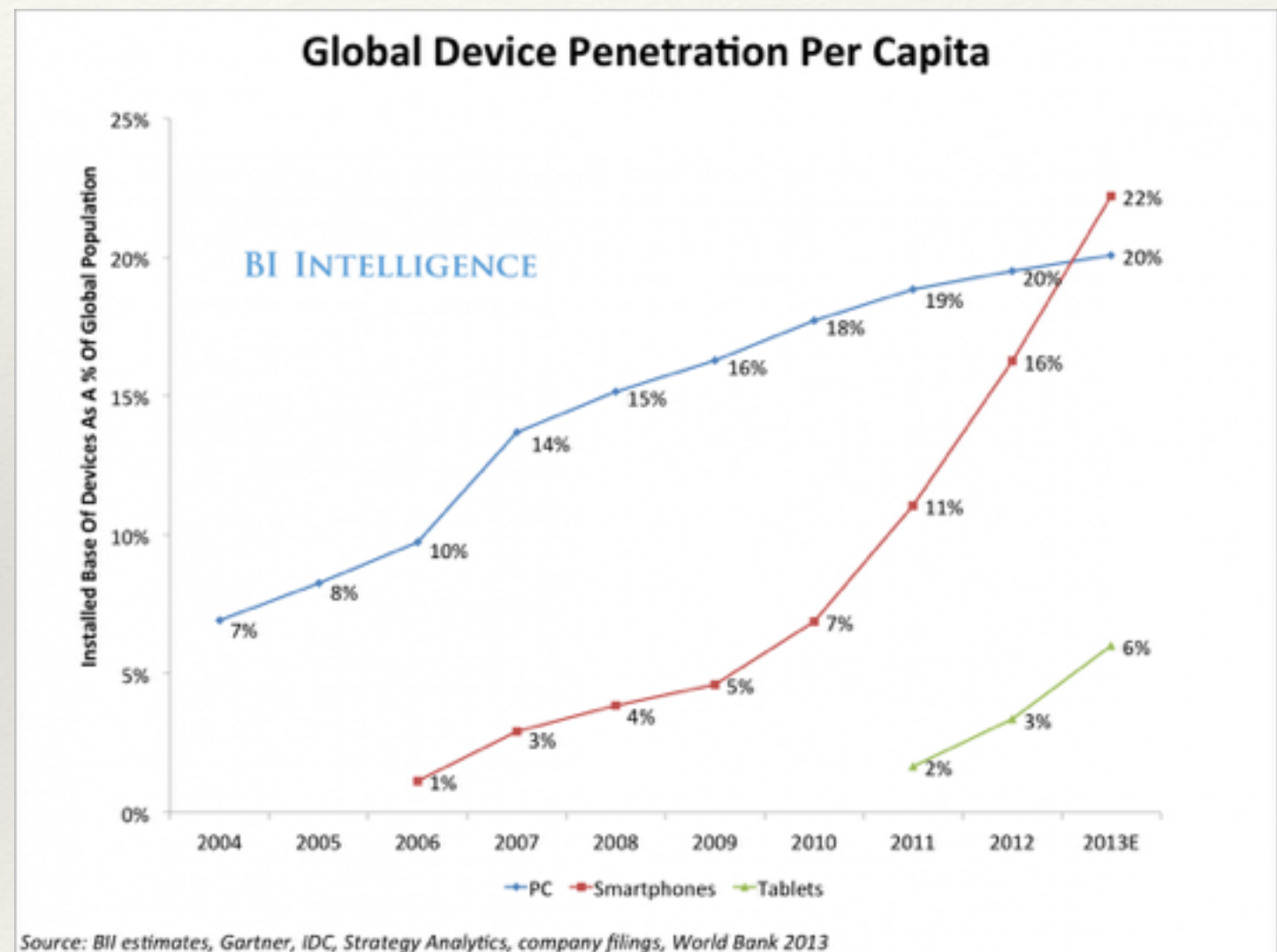
Current Electricity Generation is not so green...

2011 Total Emissions Country Rank	Country	2011 Total Carbon Dioxide Emissions from the Consumption of Energy (Million Metric Tons)
1.	China	8715.31
2.	United States	5490.63
3.	Russia	1788.14
4.	India	1725.76
5.	Japan	1180.62
6.	Germany	748.49
7.	Iran	624.86
8.	South Korea	610.95
9.	Canada	552.56
10.	Saudi Arabia	513.53
11.	United Kingdom	496.80
12.	Brazil	475.41

- ❖ ICT infrastructure generated 1.9% of CO2 Emissions in 2011
- ❖ Would be ranked as the 7th largest CO2 Emitter

Then there are mobile devices

- ❖ There are now more Smartphone in the world than PCs
- ❖ Each has a limited store of energy between charges
- ❖ How can Software Developers efficiently utilise this energy?



What is the current state of things?

- ❖ There exists a disconnect between the source code written and the energy consumed when compiled.
- ❖ Tools have been developed to guide users to inefficient areas of their software: vLens, eLens, J-RAPL, etc.
- ❖ These tools offer great guidance but leave responsibility of fixing inefficiencies to developers.
- ❖ We argue for an automated approach

Genetic Improvement

- ❖ A Search-Based Software Engineering technique
- ❖ Treats program code as if it were genetic material that can then be evolved
- ❖ Shown effective at reducing execution time.
- ❖ Effectiveness at improving other attributes is currently unknown
- ❖ Previously used to optimise MiniSAT's execution time

Research Questions

- ❖ **RQ1** To What extent can MiniSAT's Energy Consumption be reducing using Genetic Improvement?
- ❖ **RQ2** Do different downstream MiniSAT applications require different optimisations?
- ❖ **RQ3** Does reduction in energy consumption correlate to reduction in execution time when GI is applied?

Source-code

```
Lit p; int i, j;
for(i = j = 0, p = lit_Undef; i < ps.size(); i++)
{
    if(value(ps[i]) == l_True || ps[i] == ~p)
    {
        return true;
    }
    else if(value(ps[i]) != l_False && ps[i] != p)
    {
        ps[j++] = p = ps[i];
    }
}
```

Lines 105 to 116 from Solver.C

Converted to...

```
<Solver_105> ::= "Lit p; int i, j;\n"
<Solver_106> ::= "for(" <for1_Solver_106> ";" <for2_Solver_106> ";" <for3_Solver_106> ") \n"
<for1_Solver_106> ::= "i = j = 0, p = lit_Undef"
<for2_Solver_106> ::= "i < ps.size()"
<for3_Solver_106> ::= "i++"
<Solver_107> ::= "{\n"
<Solver_108> ::= " if" <IF_Solver_108> " \n"
#"if
<IF_Solver_108> ::= "(value(ps[i]) == l_True || ps[i] == ~p)"
<Solver_109> ::= "{\n"
<Solver_110> ::= " return true;\n"
<Solver_111> ::= "}\n"
<Solver_112> ::= "else if" <IF_Solver_112> " \n"
#if2
<IF_Solver_112> ::= "(value(ps[i]) != l_False && ps[i] != p)"
<Solver_113> ::= "{\n"
<Solver_114> ::= "" <_Solver_114> "\n"
#other
<_Solver_114> ::= "ps[j++] = p = ps[i];"
<Solver_115> ::= "}\n"
<Solver_116> ::= "}\n"
```

Genotype Representation

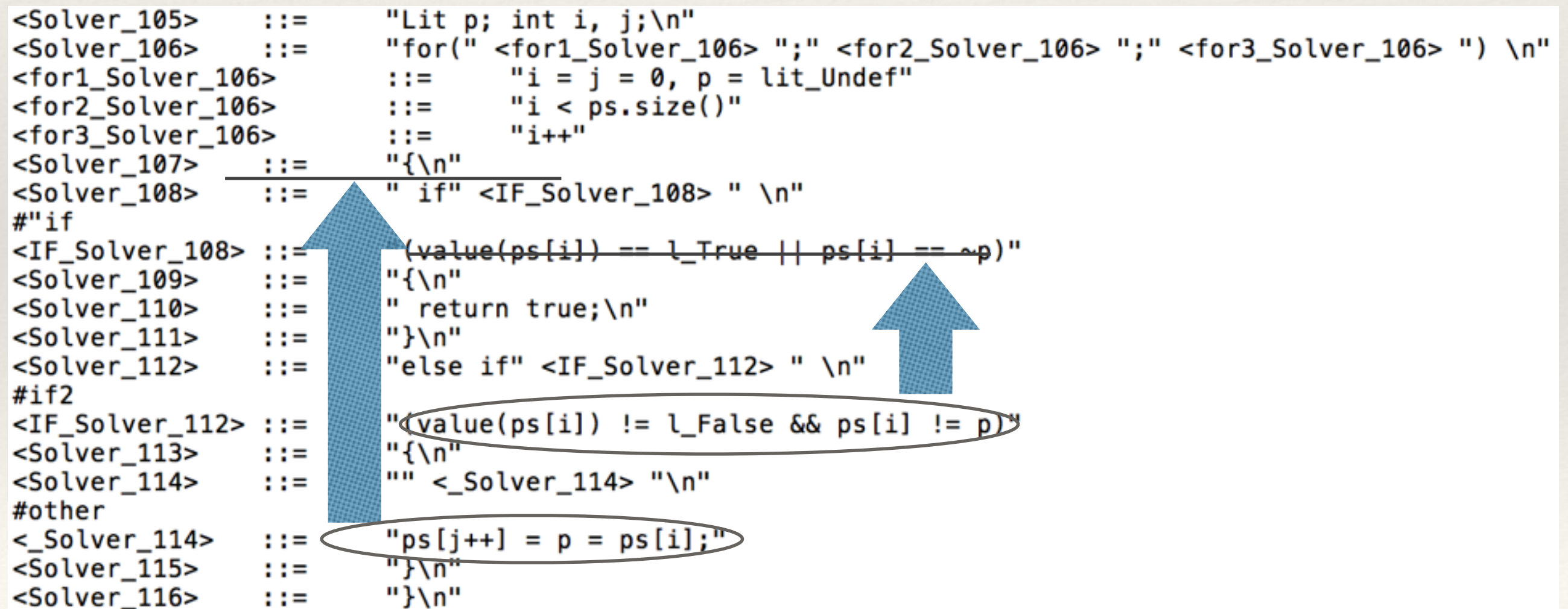
- ❖ Genotypes are simply lists of modifications made to source code. There exists three possible mutations
- ❖ DELETE: Removes a line.
<IF_Solver_108>
- ❖ COPY: Copies a line from one location to another
<Solver_108>+<_Solver_114>
- ❖ REPLACE: Replaces one line of code with another
<IF_Solver_108><IF_Solver_112>

Example Genotype

Consider the following genotype

<Solver_108>+<_Solver_114><IF_Solver_108><IF_Solver_112>

```
<Solver_105> ::= "Lit p; int i, j;\n"
<Solver_106> ::= "for(" <for1_Solver_106> ";" <for2_Solver_106> ";" <for3_Solver_106> ") \n"
<for1_Solver_106> ::= "i = j = 0, p = lit_Undef"
<for2_Solver_106> ::= "i < ps.size()"
<for3_Solver_106> ::= "i++"
<Solver_107> ::= "{\n"
<Solver_108> ::= " if" <IF_Solver_108> " \n"
#"if
<IF_Solver_108> ::= "(value(ps[i]) == l_True || ps[i] == ~p)"
<Solver_109> ::= "{\n"
<Solver_110> ::= " return true;\n"
<Solver_111> ::= "}\n"
<Solver_112> ::= "else if" <IF_Solver_112> " \n"
#if2
<IF_Solver_112> ::= "(value(ps[i]) != l_False && ps[i] != p)"
<Solver_113> ::= "{\n"
<Solver_114> ::= "" <_Solver_114> "\n"
#other
<_Solver_114> ::= "ps[j++] = p = ps[i];"
<Solver_115> ::= "}\n"
<Solver_116> ::= "}\n"
```




```

Lit p; int i, j;
for(i = j = 0, p = lit_Undef; i < ps.size(); i++)
{
    if(value(ps[i]) == l_True || ps[i] == ~p)
    {
        return true;
    }
    else if(value(ps[i]) != l_False && ps[i] != p)
    {
        ps[j++] = p = ps[i];
    }
}

```



```

Lit p; int i, j;
for(i = j = 0, p = lit_Undef; i < ps.size(); i++)
{
    ps[j++] = p = ps[i];
    if(value(ps[i]) != l_False && ps[i] != p)
    {
        return true;
    }
    else if(value(ps[i]) != l_False && ps[i] != p)
    {
        ps[j++] = p = ps[i];
    }
}

```

Fitness Function

$$Fitness = \frac{\sum_{i=1}^n energy(O, T_i)}{\sum_{i=1}^n energy(C, T_i)}$$

- ❖ For n tests (T) selected from the training set, the original Minisat (O) is compared to the candidate solution (C)
- ❖ $Fitness > 1$ indicates more energy efficient candidate
- ❖ Energy consumption estimated using Intel Power Gadget API

Selection

Genotype	Fitness	Tests Passed
<_Solver_102>	1.20	5
<IF_Solver_5><IF_Solver_13>	1.04	4
<_Solver_55><_Solver_2>	1.03	2
<IF_Solver_35>	0.98	5
<_Solver_105><_Solver_56><_Solver_102>	0.90	5
<WHILE_Solver_2><WHILE_Solver_45><_Solver_102>	0.87	4
<_Solver_6>	0.86	2
<WHILE_Solver_2><WHILE_Solver_45>	0.83	4
<_Solver_10><_Solver_54>	0.75	3
<_Solver_10><_Solver_54><_Solver_6><IF_Solver_35>	0.65	5

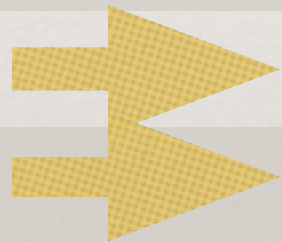
Crossover and Mutation

Genotype

< Solver_102> <IF_Solver_45><IF_Solver_67>

<IF_Solver_5><IF_Solver_13>

<IF_Solver_35> <_Solver_104>



<_Solver_102> <IF_Solver_5><IF_Solver_13>

<IF_Solver_5><IF_Solver_13> <IF_Solver_35>

<_Solver_55>+<_Solver_44>

<WHILE_Solver_101>

<_Solver_99>

<for1_Solver_144><for1_Solver_120>

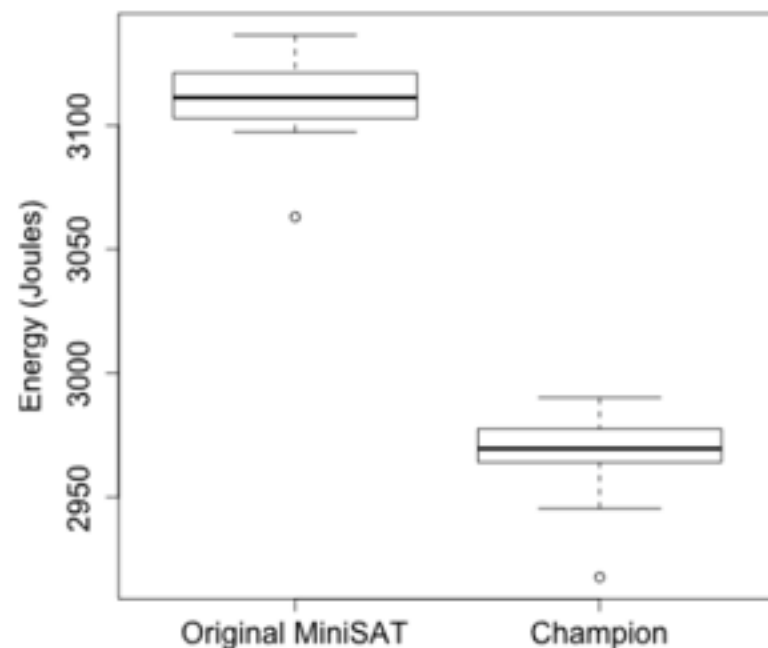
<for3_Solver_144>

Our Experiments

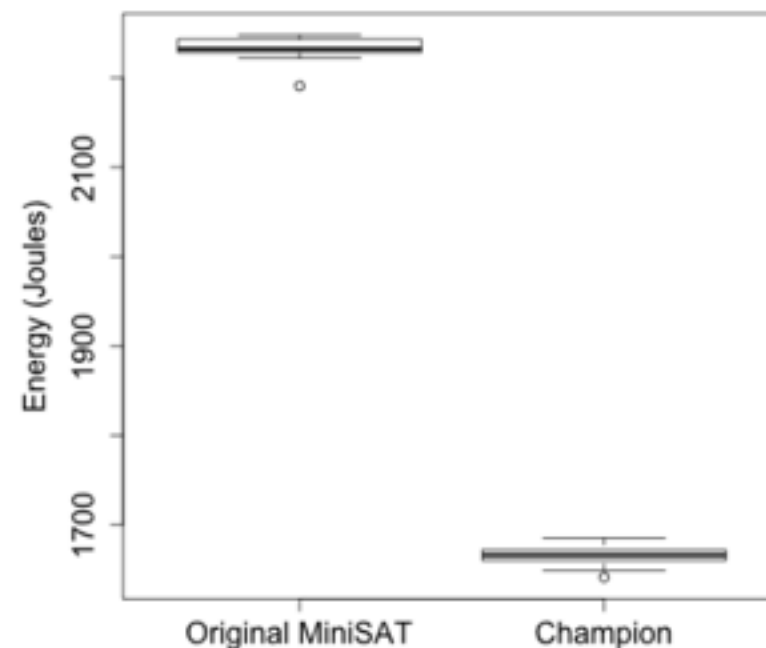
- ❖ Each Experiment optimises MiniSAT but for a different test set, representative of a unique downstream application.
- ❖ Three downstream applications: AProVE; Combinatorial Interaction Testing(CIT); Ensemble Computation of Equivalent Circuits
- ❖ Evolved 100 individuals for 20 generations

RQ1: To What extent can MiniSAT's Energy Consumption be reducing using Genetic Improvement?

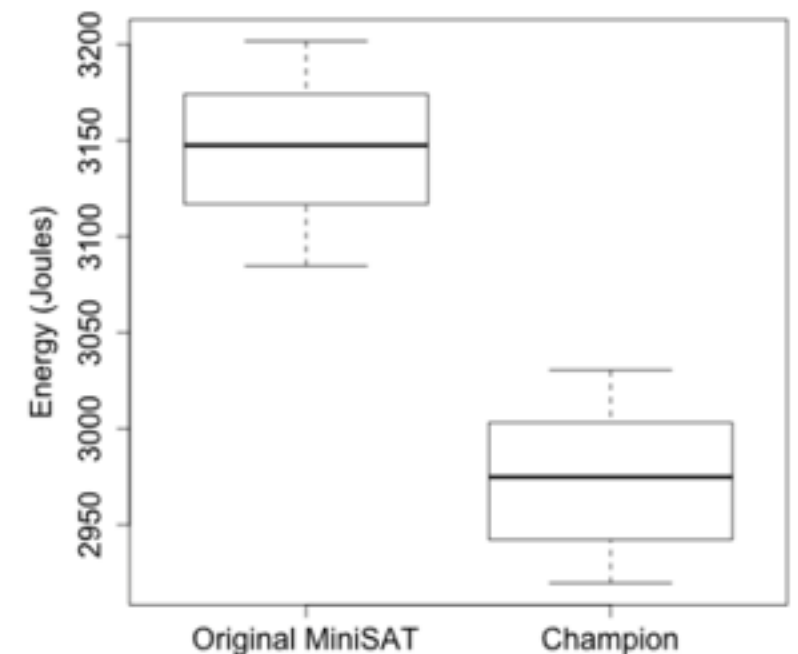
Application	Original(J)	Champ(J)	Reduction(%)
CIT	3111	2969	4.58
Ensemble	2232	1665	25.39
AProVE	3145	2973	5.44



(a) CIT



(b) Ensemble



(c) AProVE

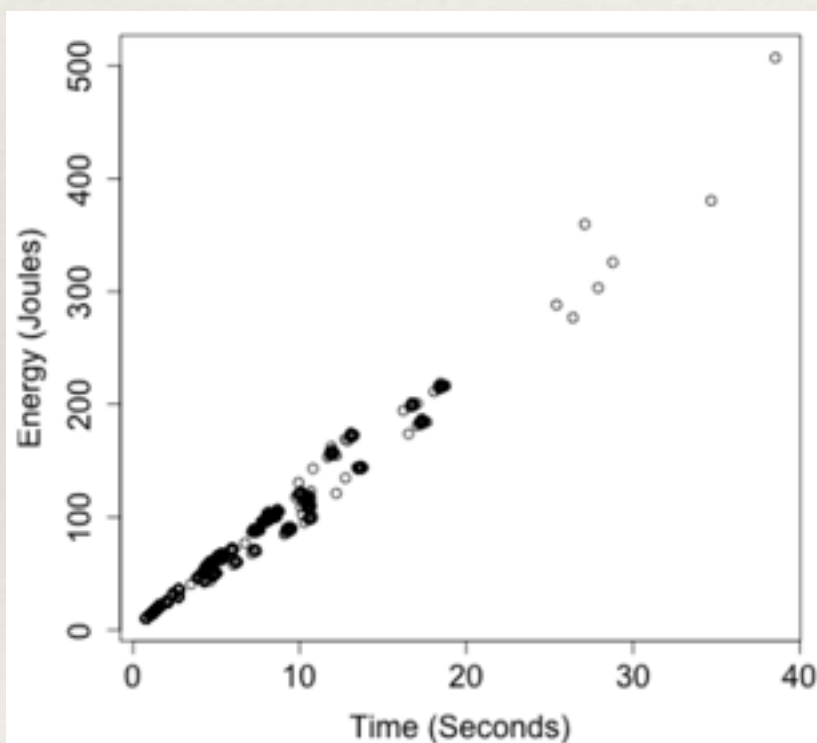
RQ2: Do different downstream MiniSAT applications require different optimisation?

-	On CIT	On Ensemble	On AProVE
CIT	-	X	X
Ensemble	X	-	X
AProVE	3.56%	3.86%	-

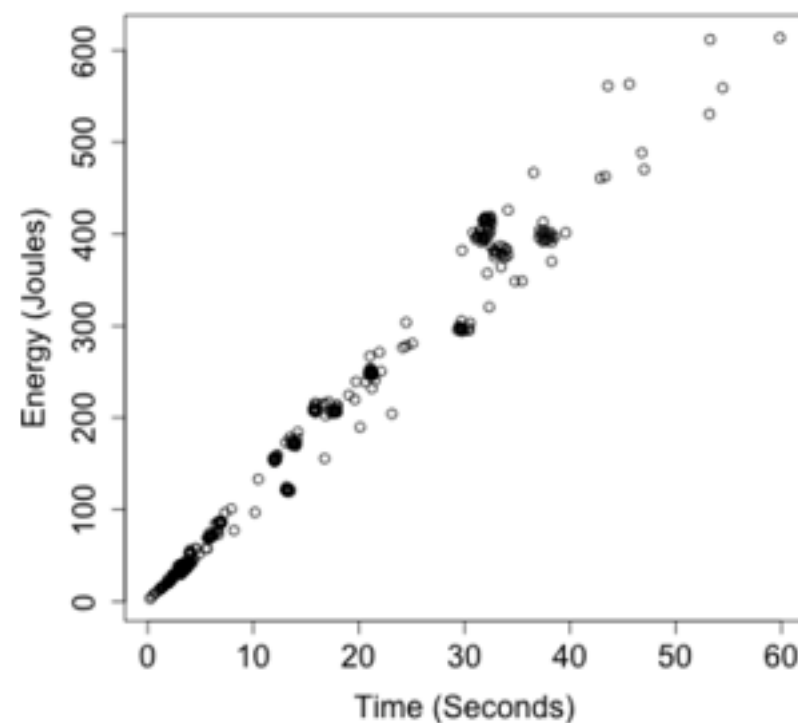
- ❖ AProVE champion solution simply removes an assert statement
- ❖ CIT champion solution disables if statement in MiniSAT's "pickBranch" function.
- ❖ Ensemble champion solution makes a change to a switch statement, equivalent to changing MiniSAT's polarity mode from polarity_false to polarity_true

RQ3: Does reduction in energy consumption correlate to reduction in execution time when GI is applied?

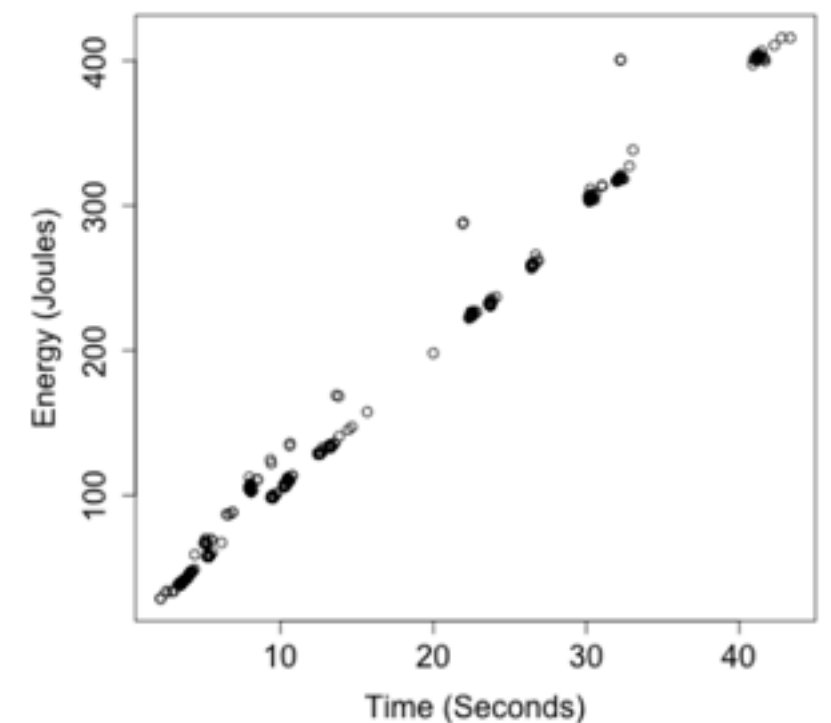
Application	Unmodified(s)	Champion(s)	Reduction
CIT	268	261	2.58%
Ensemble	219	162	25.89%
AProVE	280	261	6.69%



(a) CIT



(b) AProVE



(c) Ensemble

In Summary

- ❖ Energy consumption can be reduced by as much as 25%
- ❖ Specialised solutions found
- ❖ Some optimisations may be hard for a human to find
- ❖ Strong correlation between energy usage and execution time

