A Modularity-based Random SAT Instances Generator

Jesús Giráldez-Cru and Jordi Levy

Artificial Intelligence Research Institute
Spanish National Research Council
(**IIIA-CSIC**)
Barcelona, Spain

IJCAI 2015
July 31, 2015

Industrial SAT Instances

- The **Boolean Satisfiability Problem** (**SAT**).
- Many real-world problems are **efficiently** solved by modern SAT solvers.
  - Conflict-Driven Clause Learning (**CDCL**) SAT solvers.
- **Industrial** SAT Instances:
  - Problems encodings from real-world applications.
  - No precise definition/model: crypto, bmc, scheduling, planning, ...
  - Heterogeneity.
  - Finite and reduced number of instances.

Realistic Pseudo-Industrial SAT Instances Generators

- The **generation** of **realistic** random **pseudo-industrial** SAT instances:
  - [SelmanKautzMcAllester97]:
    CHALLENGE *10: Develop a* **generator** *for problem instances that have* **computational properties** *that are* **more similar to real world instances***.*
  - [KautzSelman03]
  - [Dechter03]
- **Need**: **Testing** and **Debugging** Purposes.
  - Desired size.
  - Desired hardness.
  - Desired properties.
- **Approach**: **Analysis** of SAT instances.
  - General common properties.
  - Isolate some of them.

Realistic Pseudo-Industrial SAT Instances Generators

- The **generation** of **realistic** random **pseudo-industrial** SAT instances:
  - [SelmanKautzMcAllester97]:
    CHALLENGE *10: Develop a* **generator** *for problem instances that have* **computational properties** *that are* **more similar to real world instances***.*
  - [KautzSelman03]
  - [Dechter03]
- **Need**: **Testing** and **Debugging** Purposes.
  - Desired size.
  - Desired hardness.
  - Desired properties.
- **Approach**: **Analysis** of SAT instances.
  - General common properties.
  - Isolate some of them.

The Community Structure of Graphs

- A graph has clear **community structure** if its nodes can be grouped into communities such that its edges mostly connect nodes of the same community.
- The **modularity** Q [NewmanGirvan04] of a graph G and a partition C of its nodes measures the *fraction of internal edges* (w.r.t. to a random graph with same nodes and same degrees).

$$Q(G,C) = \sum_{C_i \in C} \frac{\sum_{x,y \in C_i} w(x,y)}{\sum_{x,y \in V} w(x,y)} - \left(\frac{\sum_{x \in C_i} deg(x)}{\sum_{x \in V} deg(x)} \right)^2$$

  where $G = (V, w)$ and $deg(x) = \sum_{y \in V} w(x,y)$.
- The **modularity** of a graph is the **maximal** modularity for any possible partition: $Q(G) = max\{Q(G,C) | C\}$.
- The (**optimal**) modularity ranges in the interval $[0,1]$.

The Community Structure of Graphs

- A graph has clear **community structure** if its nodes can be grouped into communities such that its edges mostly connect nodes of the same community.
- The **modularity** Q [NewmanGirvan04] of a graph G and a partition C of its nodes measures the *fraction of internal edges* (w.r.t. to a random graph with same nodes and same degrees).

$$Q(G,C) = \sum_{C_i \in C} \frac{\sum_{x,y \in C_i} w(x,y)}{\sum_{x,y \in V} w(x,y)} - \left(\frac{\sum_{x \in C_i} deg(x)}{\sum_{x \in V} deg(x)} \right)^2$$

where $G = (V, w)$ and $deg(x) = \sum_{y \in V} w(x,y)$.

- The **modularity** of a graph is the **maximal** modularity for any possible partition: $Q(G) = max\{Q(G,C) | C\}$.
- The (**optimal**) modularity ranges in the interval $[0,1]$.

The Community Structure of Graphs

- A graph has clear **community structure** if its nodes can be grouped into communities such that its edges mostly connect nodes of the same community.
- The **modularity** Q [NewmanGirvan04] of a graph G and a partition C of its nodes measures the *fraction of internal edges* (w.r.t. to a random graph with same nodes and same degrees).

$$Q(G,C) = \sum_{C_i \in C} \frac{\sum_{x,y \in C_i} w(x,y)}{\sum_{x,y \in V} w(x,y)} - \left(\frac{\sum_{x \in C_i} deg(x)}{\sum_{x \in V} deg(x)} \right)^2$$

where $G = (V, w)$ and $deg(x) = \sum_{y \in V} w(x,y)$.
- The **modularity** of a graph is the **maximal** modularity for any possible partition: $Q(G) = max\{Q(G,C) | C\}$.
- The (**optimal**) modularity ranges in the interval $[0,1]$.

The Community Structure of Industrial SAT Instances

- **SAT Instances** as Graphs.
- The **Variable Incidence Graph** (**VIG**):
  - Nodes are variables.
  - Edges between two variables in the same clause.
  - Weights to give the same relevance to all clauses.

- **Industrial SAT instances** have a **clear community structure**.
- Their **modularity** has values **greater than** 0.7 in most cases (**random** SAT instances have a modularity **smaller than** 0.3).
  [AnsóteguiGiráldezLevy12]

The Community Structure of Industrial SAT Instances

- **SAT Instances** as Graphs.
- The **Variable Incidence Graph** (**VIG**):
  - Nodes are variables.
  - Edges between two variables in the same clause.
  - Weights to give the same relevance to all clauses.

- **Industrial SAT instances** have a **clear community structure**.
- Their **modularity** has values **greater than** 0.7 in most cases (**random** SAT instances have a modularity **smaller than** 0.3).
  [AnsóteguiGiráldezLevy12]

Community Attachment Generator (I)

- Classical Random k-CNF model: $F_k(n, m)$
  - n: #variables, m: #clauses, k: clause size
- **Community Attachment** model: $F_k(n, m, c, P)$
  - c: #communities (each community has n/c variables)
  - Probability P

- For *each clause*, **repeat**:
  - With probability P: all literals in the **same community**.
  - With probability $1 - P$: all literals in **distinct communities**.

  - Communities randomly chosen among c.
  - Variables randomly chosen among n/c.
  - Polarities randomly assigned.

  - Avoiding trivial clauses (repeated literals or tautologies).

- Restriction: $k \leq c \leq n/k$
  - There always exists at least one possible clause to select.

Community Attachment Generator (I)

- Classical Random k-CNF model: $F_k(n, m)$
  - n: #variables, m: #clauses, k: clause size
- **Community Attachment** model: $F_k(n, m, c, P)$
  - c: #communities (each community has n/c variables)
  - Probability P

- For *each clause*, **repeat**:
  - With probability P: all literals in the **same community**.
  - With probability $1 - P$: all literals in **distinct communities**.

  - Communities randomly chosen among c.
  - Variables randomly chosen among n/c.
  - Polarities randomly assigned.

  - Avoiding trivial clauses (repeated literals or tautologies).

- Restriction: $k \leq c \leq n/k$
  - There always exists at least one possible clause to select.

Community Attachment Generator (I)

- Classical Random k-CNF model: $F_k(n, m)$
  - n: #variables, m: #clauses, k: clause size
- **Community Attachment** model: $F_k(n, m, c, P)$
  - c: #communities (each community has n/c variables)
  - Probability P

- For *each clause*, **repeat**:
  - With probability P: all literals in the **same community**.
  - With probability $1 - P$: all literals in **distinct communities**.

  - Communities randomly chosen among c.
  - Variables randomly chosen among n/c.
  - Polarities randomly assigned.

  - Avoiding trivial clauses (repeated literals or tautologies).

- Restriction: $k \leq c \leq n/k$
  - There always exists at least one possible clause to select.

Community Attachment Generator (I)

- Classical Random k-CNF model: $F_k(n, m)$
  - n: #variables, m: #clauses, k: clause size
- **Community Attachment** model: $F_k(n, m, c, P)$
  - c: #communities (each community has n/c variables)
  - Probability P

- For *each clause*, **repeat**:
  - With probability P: all literals in the **same community**.
  - With probability $1 - P$: all literals in **distinct communities**.

  - Communities randomly chosen among c.
  - Variables randomly chosen among n/c.
  - Polarities randomly assigned.

  - Avoiding trivial clauses (repeated literals or tautologies).

- Restriction: $k \leq c \leq n/k$
  - There always exists at least one possible clause to select.

Community Attachment Generator (II)

- **Community Attachment** model: $F_k(n, m, c, P)$

- **Theorem.** Given a SAT instance $\Gamma \in F_k(n, m, c, P)$, let G be its VIG. The **average modularity** of G is bounded as:

$$E[Q(G)] \geq P - \frac{1}{c}$$

- When P is **big enough**, the modularity is very close to this lower-bound. Therefore, we simply take:

$$P = Q + \frac{1}{c}$$

Community Attachment Generator (II)

- **Community Attachment** model: $F_k(n, m, c, P)$

- **Theorem.** Given a SAT instance $\Gamma \in F_k(n, m, c, P)$, let G be its VIG. The **average modularity** of G is bounded as:

$$E[Q(G)] \geq P - \frac{1}{c}$$

- When P is **big enough**, the modularity is very close to this lower-bound. Therefore, we simply take:

$$P = Q + \frac{1}{c}$$

Community Attachment Generator (II)

- **Community Attachment** model: $F_k(n, m, c, P)$

- **Theorem.** Given a SAT instance $\Gamma \in F_k(n, m, c, P)$, let G be its VIG. The **average modularity** of G is bounded as:

$$E[Q(G)] \geq P - \frac{1}{c}$$

- When P is **big enough**, the modularity is very close to this lower-bound. Therefore, we simply take:

$$P = Q + \frac{1}{c}$$

Community Attachment Generator (III)

- **High** modularity:
  more adequate to model **real-world** problems.

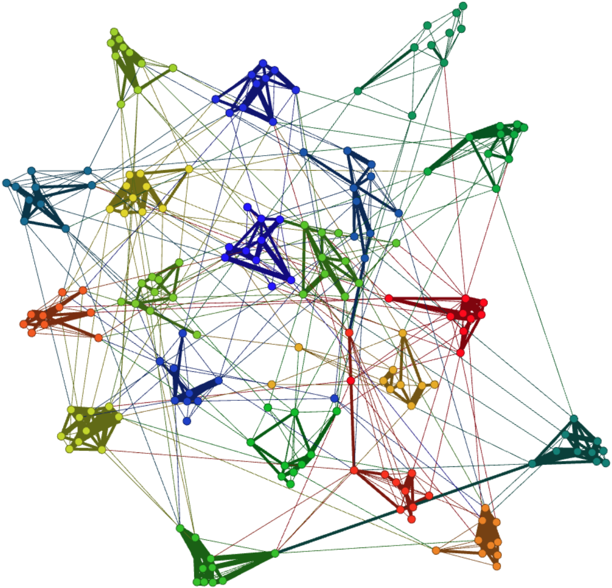
- **Low** modularity:
  more similar to classical **random** problems.

This generator is publicly available at:

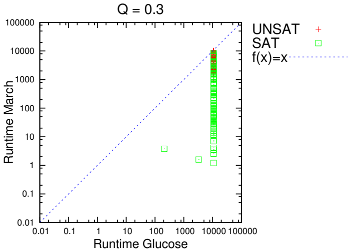
`http://www.iiia.csic.es/~jgiraldez/software`

Example

200 variables
850 clauses
$Q = 0.8$
20 communities

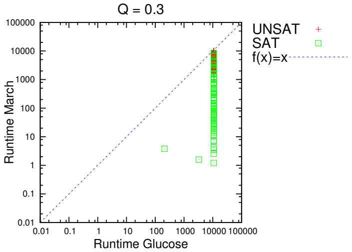


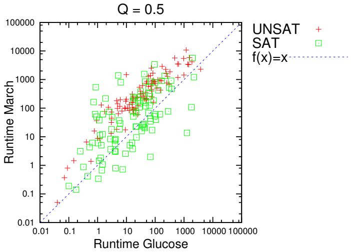
Glucose (*industrial*) vs **March** (*random*):



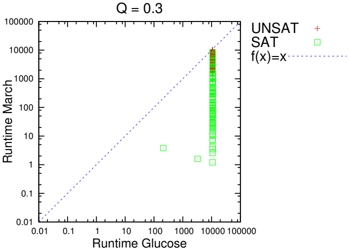
SAT Solvers Performance

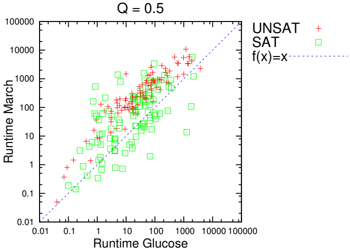
Glucose (*industrial*) vs **March** (*random*):

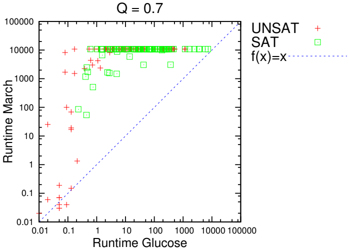




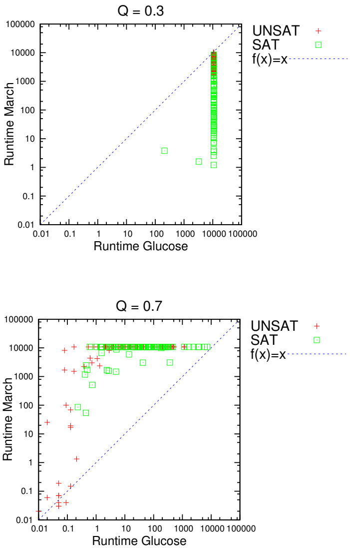
Glucose (*industrial*) vs **March** (*random*):





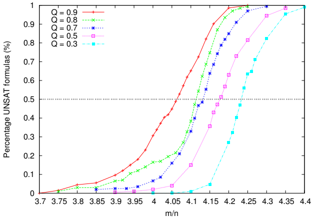


Glucose (*industrial*) vs **March** (*random*):



The Phase Transition Point

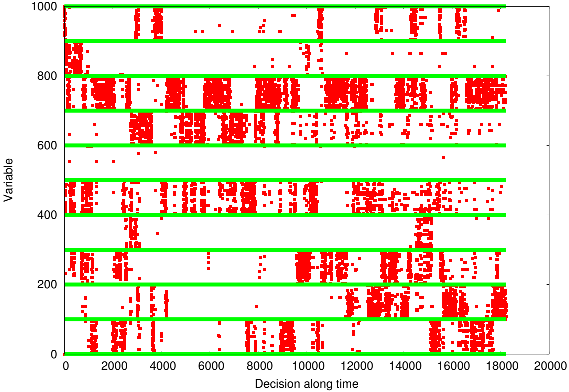
- Finite size (*Observations*):
  The **phase transition point SAT-UNSAT**, *if exists*, **decreases** for higher values of **modularity**.



- Infinite size (*Theorem*):
  The **phase transition point SAT-UNSAT**, *if exists*, does **not depend** on the **modularity.**
  - **Proof**.

Analyzing the SAT Solver

- 1000 variables and 10 communities.



- Using community structure to improve the performance of CDCL SAT solvers: [AnsóteguiGiráldezLevySimon] to *appear* in **SAT 2015**.

Thank you for your attention!

Poster Panel #49

I am looking for postdocs opportunities

jgiraldez@iiia.csic.es