Formalization of the Domination Chain with Weighted Parameters

Daniel E. Severín
Depto. de Matemática, Universidad Nacional de Rosario, Argentina
CONICET, Argentina
http://www.fceia.unr.edu.ar/~daniel
daniel@fceia.unr.edu.ar

Abstract

The *Cockayne-Hedetniemi Domination Chain* is a chain of inequalities between classic parameters of graph theory: for a given graph G, $ir(G) \leq \gamma(G) \leq \iota(G) \leq \alpha(G) \leq \Gamma(G) \leq IR(G)$. These parameters return the maximum/minimum cardinality of a set satisfying some property. However, they can be generalized for graphs with weighted vertices where the objective is to maximize/minimize the sum of weights of a set satisfying the same property, and the domination chain still holds for them. In this work, the definition of these parameters as well as the chain is formalized in Coq/Ssreflect.

2012 ACM Subject Classification Mathematics of computing → Graph theory

Keywords and phrases Domination Chain, Coq, Formalization of Mathematics

Digital Object Identifier 10.4230/LIPIcs.ITP.2019.36

Category Short Paper

Supplement Material The Coq formalization and the solver accompanying this paper can be found at https://dx.doi.org/10.17632/h5j5rvrz2r.2.

Funding Partially supported by grants ANPCyT PICT-2016-0410 and PID-UNR ING538.

Acknowledgements I want to thank Ricardo Katz for his careful reading and suggestions.

1 Introduction

The domination parameters and the relationship between them is a very active research area due to the numerous applications that can be modeled with them. They are introduced below, following the treatment given in the textbook [9].

Let $G = (V, E)$ be a simple graph. For any $v \in V$, let $N(v)$ be the set of vertices adjacent to v and $N[v] \doteq N(v) \cup \{v\}$. For any $S \subseteq V$, let $N(S) \doteq \bigcup_{v \in S} N(v)$ and $N[S] \doteq \bigcup_{v \in S} N[v]$.

A set $S \subseteq V$ is called a *stable set* if $N(S) \cap S = \emptyset$. Alternatively, S is a stable set if no vertex in S is adjacent to any other vertex in S. The *independence number* $\alpha(G)$ of a graph G is the maximum cardinality of a stable set in G.

A set $D \subseteq V$ is called a *dominating set* if $N[D] = V$. Alternatively, D is a dominating set if for all $v \in V - D$, there exists a vertex $u \in D$ such that u is adjacent to v. The *domination number* $\gamma(G)$ of a graph G is the minimum cardinality of a dominating set in G and the *independence domination number* $\iota(G)$ is the minimum cardinality of a set which is stable and dominating simultaneously.

A property p is called *hereditary* if whenever a set S satisfies p, so does every proper subset $S' \subset S$. Analogously, p is called *superhereditary* if whenever a set S satisfies p, so does every proper superset $S' \supset S$. "To be stable" is an hereditary property while "to be dominating" is superhereditary.

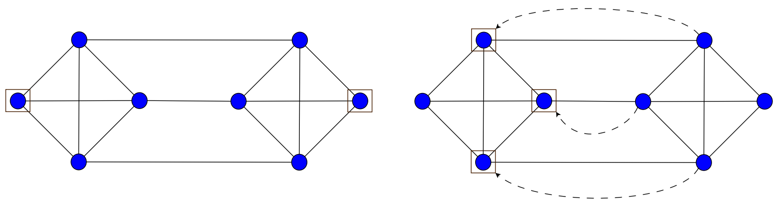

Figure 1 $\Gamma(G) = 2 < 3 = IR(G)$.

A set $S \subseteq V$ satisfying an hereditary property p is *maximal* if, for every $v \in V - S$, $S \cup \{v\}$ does not satisfy p. Similarly, a set S satisfying a superhereditary property p is *minimal* if, for every $v \in S$, $S - \{v\}$ does not satisfy p. For instance, a *minimal dominating set* D is a dominating set such that any proper subset of D is not dominating. Note that a dominating set of minimum cardinality is, in particular, minimal. Since finding $\gamma(G)$ is an NP-Hard problem, heuristics approaches to address them are usual and, in particular, a greedy heuristic consisting of adding elements to a set until it becomes dominating is one of these approaches. Such heuristic always returns a minimal dominating set by definition. Its worst case leads to the definition of the *upper domination number* $\Gamma(G)$ which is the maximum cardinality of a minimal dominating set in G.

For a given set $D \subset V$ and vertex $v \in D$, let $s_D(v) \doteq N[v] - N[D - \{v\}]$. This set has those vertices only dominated by v, whose are called *private vertices* of v in D. A set $D \subseteq V$ is called an *irredundant set* if, for every $v \in D$, $s_D(v) \neq \emptyset$. In other words, each vertex of D must dominate at least one vertex not dominated by any other vertex from D. The *upper irredundance number* $IR(G)$ is the maximum cardinality of an irredundant set in G. "To be irredundant" is an hereditary property and, thus, one might be interested in finding the minimum cardinality of a maximal irredundant set. The latter is called the *lower irredundance number* and denoted by $ir(G)$.

Figure 1 shows an example of a minimal dominating set (on the left) and an irredundant set (on the right). Both sets are represented by vertices inside boxes. In the right graph, arrows link vertices from the irredundant set to their private vertices. This graph is a known example where Γ and IR differs [10].

According to Favaron et al. [6], more that 1500 research papers about dominating sets have been published and, in particular, more than 100 explore properties of irredundant sets in graphs, showing the importance of this topic (which is still active [2]). Despite that, and to the best of my knowledge, these concepts have not been formalized yet.

This ongoing work intends to reduce the gap between what is already informally proved and what is not, such that other graph theorists may have a framework to formalize their results, especially when their proofs require the analysis of dozens of mechanical cases (the Four-Color Theorem is an example of a result involving an overwhelming number of cases [7]). In particular, this work is the basis to prove later that IR_w (defined in the next section) is polynomial on $\{claw, bull, P_6, \overline{C_6}\}$-free graphs [12]. However, it requires to consider several "boring" cases and its formalization could be a way to channel this result, reaching a twofold goal: on the one hand, to get confident about the proof and, on the other, to have the advantage that a reader can accept it without the need of manually checking step by step (it eventually could reduce the time spent in the peer-review process).

Another line of research that motivated this work is presented at the end of the paper.

A starting point is to formalize in Coq/Ssreflect [8] the Cockayne-Hedetniemi domination chain, which is the basis for many other results [9]. It states that for any graph G,

$$ir(G) \leq \gamma(G) \leq \iota(G) \leq \alpha(G) \leq \Gamma(G) \leq IR(G).$$

The proof relies on the following facts:

- A stable set D is maximal if and only if D is stable and dominating (see Prop. 3.5 of [9]).
- A maximal stable set is a minimal dominating set (see Prop. 3.6 of [9]).
- A dominating set D is minimal if and only if D is dominating and irredundant (see Prop. 3.8 of [9]).
- A minimal dominating set is a maximal irredundant set (see Prop. 3.9 of [9]).

For instance, in order to prove $ir(G) \leq \gamma(G)$ one can pick a dominating set D of minimum cardinality, i.e. $|D| = \gamma(G)$. Since D is minimal dominating, it is also maximal irredundant. Therefore, $|D| \geq ir(G)$.

2 Weighted parameters

The parameters defined in the previous section can be generalized as follows. For a given graph $G = (V, E)$, consider a positive integer weight $w(v)$ associated to each vertex v, i.e. $w : V \to \mathbb{N}_1$, where $\mathbb{N}_1$ denotes the set of natural numbers starting from 1. For any $S \subseteq V$, define the weight of S as $w(S) \doteq \sum_{v \in S} w(v)$. Let $\beta \in \{ir, \gamma, \iota, \alpha, \Gamma, IR\}$ be a parameter consisting of minimizing (or maximizing) the cardinality of a set S satisfying the corresponding property p (e.g. if $\beta = \alpha$ then the objective is "to maximize" and p is "to be a stable set"), and define $\beta_w(G)$ as the value of $w(S)$ such that S satisfies p and minimizes (maximizes resp.) $w(S)$.

Since weights are positive, sets of minimum (maximum resp.) weight are also minimal (maximal resp.), and the domination chain still holds for these parameters:

▶ **Theorem 1.** *For any graph G and weights $w : V(G) \to \mathbb{N}_1$, $ir_w(G) \leq \gamma_w(G) \leq \iota_w(G) \leq \alpha_w(G) \leq \Gamma_w(G) \leq IR_w(G)$.*

In particular, the problems of finding $\gamma_w(G)$ and $\alpha_w(G)$ are the classic optimization problems Minimum Weighted Dominating Set and Maximum Weighted Stable Set. Nevertheless, the weighted versions of the remaining parameters are also beginning to be studied: some theoretical results about $\Gamma_w(G)$ have recently been reported [3] and algorithms for obtaining (a generalized form of) $\iota_w(G)$ have been proposed [4].

Therefore, it makes sense to directly formalize the domination chain for the weighted case, and the original chain can be proved straightforwardly by setting $w(v) = 1$ for all $v \in V$. The code accompanying this paper (from now on, *the code*) contains 3 Coq files, described below:

Name	Definitions	Proofs	Lines (spec)	Lines (proof)
`basics.v`	12	46	282	303
`dom.v`	55	60	311	551
`example.v`	1	17	62	166

The second and third column display the number of global definitions and proofs, and the fourth and fifth column show the number of lines of specification and proof reported by the tool `coqwc`. The total number of lines (spec + proof) amounts to 1675. Also, there is a browsable version of the code made with *CoqDocJS*, and a solver for computing parameters γ_w, ι_w, α_w, Γ_w and IR_w. The solver can also generate a Coq file with a proof of $\alpha(G) \geq k$.

3 Graph definition

This section is devoted to briefly discussing how to represent a finite simple graph in the language Coq. First, a description of current representations is given.

The Mathematical Components library [13] (from now on, *MC library*) is equipped with a definition of finite graphs, which can be consulted in the file `fingraph.v`. Basically, vertices are elements of a finite type T and a graph is represented by a function of type $T \to$ `seq` T, i.e. an assignment from vertices to lists containing their adjacencies (here, `seq` is the ssreflect type for sequences, see `seq.v` from [13]). This library also has some basic results about connectivity, which is seen as the transitive closure of the adjacency relation, and they are used in the formal proof of the Four-Color Theorem [7] (at that time the file was called `connect.v`).

Recently, another representation was given by Dockzal, Combette and Pous [5] since `fingraph` in the MC library as well as other results in the formal proof of the Four-Color Theorem were conceived to deal with planar graphs and do not fulfill some requirements needed for a general theory of graphs. The authors define a graph G as a structure $\langle V, R\rangle$ (called `sgraph`) where V is a finite type inhabited by the vertices of G and $R : V \to V \to$ `bool` is a symmetric and irreflexive relation representing the adjacency relation of G (and denoted by "`--`"). Several results about connectivity, morphisms, minor relation and treewidth among others are formalized.

In this work, the latter representation is adopted. Moreover, the code is compatible with the one proposed in [5] and it can certainly extend that library.

Formalizations of some aspects of graph theory are not restricted to the language Coq. One of them is the work of Noschinski [11] for Isabelle/HOL. He defines a simple graph (not necessarily finite) as a pair $(V, E) \in \mathcal{P}(\mathbb{N}) \times \mathcal{P}(\mathcal{P}(\mathbb{N}))$ (where $\mathcal{P}(X)$ denotes the powerset of X) satisfying the condition $\forall\, e \in E \bullet e \subseteq V \wedge |e| = 2$. As this set-theoretic representation can be more intuitive for newcomers, the file `basics.v` (from the code) defines the edge set $E(G)$ in terms of the adjacency relation. Some results are then expressed with $E(G)$, including one of the first classic facts given in textbooks: the sum of the degrees of all vertices is equal to twice the number of edges.

```
Theorem sumdeg_2E : ∀ G : sgraph, 2 * #|E(G)| = Σ(w in V(G)) deg G w.
```

The file `basics.v` also has simple results about finite sets and summations not found in the MC library, and definitions of open and closed neighborhoods, and the degree of vertices.

4 Formalizing the domination chain

This section exposes the most relevant details about the formalization performed in the file `dom.v`, which contains definitions and results about: 1) stable, dominating and irredundant sets, 2) private sets, 3) hereditary and superhereditary properties, 4) maximal and minimal sets, 5) sets of maximum and minimum weights, and 6) weighted and unweighted parameters.

One of the obstacles found was that it was easier to prove statements about properties over sets when they were defined as `Prop`-terms rather than `bool`-terms, while the MC library commonly uses the latter. For that reason, the concept of *property* was packaged in a structure, where a property p comes in two flavors: `vsbool`, which is a compact definition of p having type {`set G`} $\to$ `bool` (see the definition of `pred` in the MC library), and `vsprop`, which is the same property written in terms of quantifiers and having type {`set G`} $\to$ `Prop`, where {`set G`} denotes the type of sets of vertices:

```
Record vsproperty := VertexSetProperty {
  vsprop :> {set G} → Prop ;
  vsbool :  pred {set G} ;
  vsrefl :  ∀ D : {set G}, reflect (vsprop D) (vsbool D) ;
  vsinhb :  {set G} ;
  vspinh :  vsprop vsinhb
}.
```

A boolean reflection view `vsrefl` is used to prove the equivalence between the two. In addition, the structure is equipped with a set `vsinhb` satisfying the property p. Its proof is given in `vspinh`. For instance, stable sets are defined as follows:

```
Definition stable := @VertexSetProperty
       p_stable pb_stable stableP ∅ st_empty.
```

where `p_stable` and `pb_stable` are the two versions given below, `stableP` is the reflection view between them, and `st_empty` is a proof that the empty set is stable.

```
Definition p_stable := ∀ u v : G, u ∈ S → v ∈ S → ¬ (u -- v).
Definition pb_stable := NS(S) ∩ S == ∅.
```

In the code, `NS(S)` is notation for $N(S)$, i.e. the open neighborhood of a set S.

Having different definitions in both types is useful and has already been applied previously in the MC library: for example, the lemma `set0Pn` proves the equivalence between the `Prop`-term "`∃ x, x ∈ A`" and the `bool`-term "`A != ∅`" in the file `finset.v`. As it was pointed out previously, `vsbool` is mainly used when interacting with the MC library while `vsprop` is preferred for performing proofs. A coercion between `vsproperty` and `vsprop` is declared since the latter is used intensively and improves readability.

For a given property p and a given set of vertices D, the latter is a maximal set if it satisfies p but no proper superset F of D does. In addition, if p is hereditary, it is possible to apply the definition of maximal set given in the introduction (here called `maximal_altdef`):

```
Definition maximal := p D ∧ (∀ F : {set G}, D ⊂ F → ¬ p F).
Definition hereditary := ∀ F : {set G}, F ⊆ D → p D → p F.
Theorem maximal_altdef : hereditary p →
       (maximal ↔ (p D ∧ (∀ v : G, v ∉ D → ¬ p (D ∪ {v})))).
```

Something similar is done for the definitions of minimal and superhereditary. From now on, only concepts related to maximal sets are presented (keeping in mind that the same is done for minimal ones).

In order to define the property that a given set is maximal irredundant, it is required to propose an inhabitant of that property. The code gives a tool called `ex_maximal` for providing these kind of sets. For instance, `ex_maximal irredundant` generates a maximal irredundant set and `maximal_exists` gives a proof that the generated set satisfies that property.

Let p be a property, $F \doteq$ `vsinhb p`, i.e. a set satisfying p, and $pb \doteq$ `vsbool p`, i.e. the `bool`-version of the property. The following function provides a set of maximum weight:

```
Definition maximum_set := [arg max_(D > F | pb D) weight_set D].
```

where `weight_set` is the weight of a given set. Note that `maximum_set` is defined in terms of [`arg max_`$(D > F \mid P)$ M], a function from the MC library that returns an object D maximizing M subject to P, where P holds for F.

Now, we have all the elements to introduce the parameters. For instance, $IR_w(G)$ is defined as the weight of the irredundant set of maximum weight.

```
Definition IR_w := weight_set weight (maximum_set weight irredundant).
```

For unweighted cases, cardinality is used. That is:

```
Definition maximum_set_card := [arg max_(D > F | pb D) #|D|].
Definition IR := #|maximum_set_card irredundant|.
```

Then, the equivalence between both cases (when weights are `ones`) is established:

```
Lemma IR_is_IR1 : IR = IR_w ones.
```

Finally *Theorem 1* is proved and the original chain is derived as a consequence of that theorem. For instance, for the statements $\Gamma_w(G) \leq IR_w(G)$ and $\Gamma(G) \leq IR(G)$ we have:

```
Theorem Gamma_w_leq_IR_w : ∀ (G : sgraph) (weight : G → nat),
       Γ_w G weight ≤ IR_w G weight.
Corollary Gamma_leq_IR : ∀ G : sgraph, Γ G ≤ IR G.
```

The file `example.v` shows an example on how to use these concepts: it proves that a complete graph K satisfies $\alpha_w(K) = \Gamma_w(K) = IR_w(K) = \max\{w(v) : v \in V(K)\}$, by bounding $\alpha_w(K)$ from below and $IR_w(K)$ from above, and applying Theorem 1 for collapsing the three parameters. It is also shown that $ir(K) = \gamma(K) = \iota(K) = \alpha(K) = \Gamma(K) = IR(K) = 1$.

A future research line related to this work conceives the idea of obtaining the proof (as a Coq file) of the value of a parameter over instances of reasonable size. For example, suppose that a certain application is modeled as a Maximum Stable Set Problem and, after all, one wants to verify $\alpha(G) = k$, for some G and k. Certifying that $\alpha(G) \geq k$ is easy (i.e. polynomial in the size of G): propose a set of k vertices and prove that it is stable. In fact, this is done by the solver provided in the supplement material. Therefore, the effort should be put in the generation of a proof of $\alpha(G) \leq k$ as small as possible. Below, a technique is briefly elaborated. Consider an Integer Linear Programming formulation that models the problem, e.g. maximize $\sum_{i \in V(G)} x_i$ subject to $x_i + x_j \leq 1$ for all $i, j \in E(G)$ and $x_i \in \{0, 1\}$ for all i. By adding the contraint $\sum_{i \in V(G)} x_i \geq k + 1$, the formulation turns infeasible. Next, find an Irreducible Infeasible Subsystem (IIS). There are tools that perform this task, e.g. *Conflict Refiner* of IBM CPLEX (or, even better, one can get the *minimum* IIS by solving a set covering problem). Then, solve the IIS via Branch-and-Bound (the number of explored nodes can be reduced by using a *strong branching* strategy). It generates a tree where each leaf corresponds to a infeasible Linear Programming (LP) problem. Hence, the proof of $\alpha(G) \leq k$ consists mainly of enumerating these LP problems and certifying that each one is infeasible, which can be done via Farkas Lemma. The library of formalized LP concepts provided in [1] might be useful here. Integer LP formulations for γ_w, ι_w and α_w are well-known, while recent ones for Γ_w and IR_w have been proposed [12] and implemented (see supplement material).

References

1. X. Allamigeon and R. Katz. A Formalization of Convex Polyhedra Based on the Simplex Method. *J. Autom. Reasoning*, pages 1–23, 2018. doi:10.1007/s10817-018-9477-1.
2. C. Bazgan, L. Brankovic, K. Casel, H. Fernau, K. Jansen, K.-M. Klein, M. Lampis, M. Liedloff, J. Monnot, and V. Th. Paschos. The many facets of upper domination. *Theor. Comput. Sci.*, 717:2–25, 2018. doi:10.1016/j.tcs.2017.05.042.
3. A. Boyaci and J. Monnot. Weighted upper domination number. *Electron. Notes Discrete Math.*, 62:171–176, 2017. doi:10.1016/j.endm.2017.10.030.

4 P. P. Davidson, C. Blum, and J. Lozano. The weighted independent domination problem: Integer linear programming models and metaheuristic approaches. *Eur. J. Oper. Res.*, 265:860–871, 2018. doi:10.1016/j.ejor.2017.08.044.

5 C. Doczkal, G. Combette, and D. Pous. A Formal Proof of the Minor-Exclusion Property for Treewidth-Two Graphs. *Lect. Notes Comput. Sc.*, 10895:178–195, 2018. doi:10.1007/978-3-319-94821-8_11.

6 O. Favaron, T. Haynes, S. Hedetniemi, M. Henning, and D. Knisley. Total irredundance in graphs. *Discrete Math.*, 256:115–127, 2002. doi:10.1016/S0012-365X(00)00459-3.

7 G. Gonthier. Formal proof - the Four-Color Theorem. *Notices Amer. Math. Soc.*, 55:1382–1393, 2008. URL: http://www.ams.org/notices/200811/tx081101382p.pdf.

8 G. Gonthier and A. Mahboubi. An introduction to small scale reflection in Coq. *J. Form. Reason.*, 3:95–152, 2010. doi:10.6092/issn.1972-5787/1979.

9 T. Haynes, S. Hedetniemi, and P. Slater. *Fundamentals of Domination in Graphs*. Marcel Dekker, Inc., 1998.

10 M. Jacobson and K. Peters. Chordal graphs and upper irredundance, upper domination and independence. *Discrete Math.*, 86:59–69, 1990. doi:10.1016/0012-365X(90)90349-M.

11 L. Noschinski. Proof Pearl: A Probabilistic Proof for the Girth-Chromatic Number Theorem. *Lect. Notes Comput. Sc.*, 7406:393–404, 2012. doi:10.1007/978-3-642-32347-8_27.

12 D. Severín and G. Nasini. The Maximum Weighted Upper Irredundance Problem (in Spanish). In *Communications of the Unión Matemática Argentina*. UNLP, La Plata, September 2018.

13 The Mathematical Components team. Mathematical components, 2018. URL: http://math-comp.github.io/math-comp/.