# A Probabilistic Proof of the Girth-Chromatic Number Theorem

Lars Noschinski

February 23, 2021

**Abstract**

This works presents a formalization of the Girth-Chromatic number theorem in graph theory, stating that graphs with arbitrarily large girth and chromatic number exist. The proof uses the theory of Random Graphs to prove the existence with probabilistic arguments and is based on [1].

## Contents



**theory** *Girth-Chromatic-Misc*
**imports**
*Main*
*HOL−Library.Extended-Real*
**begin**

# 1 Auxilliary lemmas and setup

This section contains facts about general concepts which are not directly connected to the proof of the Chromatic-Girth theorem. At some point in time, most of them could be moved to the Isabelle base library.

Also, a little bit of setup happens.

## 1.1 Numbers

**lemma** *enat-in-Inf*:
  **fixes** $S :: enat\ set$
  **assumes** $Inf\ S \neq top$
  **shows** $Inf\ S \in S$
⟨*proof*⟩

**lemma** *enat-in-INF*:
  **fixes** $f :: 'a \Rightarrow enat$
  **assumes** $(INF\ x{\in}\ S.\ f\ x) \neq top$
  **obtains** $x$ **where** $x \in S$ **and** $(INF\ x{\in}\ S.\ f\ x) = f\ x$
⟨*proof*⟩

**lemma** *enat-less-INF-I*:
  **fixes** $f :: 'a \Rightarrow enat$
  **assumes** *not-inf*: $x \neq \infty$ **and** *less*: $\bigwedge y.\ y \in S \Longrightarrow x < f\ y$
  **shows** $x < (INF\ y{\in}S.\ f\ y)$
  ⟨*proof*⟩

**lemma** *enat-le-Sup-iff*:
  $enat\ k \leq Sup\ M \longleftrightarrow k = 0 \vee (\exists\, m \in M.\ enat\ k \leq m)$ (**is** $?L \longleftrightarrow ?R$)
⟨*proof*⟩

**lemma** *enat-neq-zero-cancel-iff*[*simp*]:
  $0 \neq enat\ n \longleftrightarrow 0 \neq n$
  $enat\ n \neq 0 \longleftrightarrow n \neq 0$
⟨*proof*⟩

**lemma** *natceiling-lessD*: $nat(ceiling\ x) < n \Longrightarrow x < real\ n$
⟨*proof*⟩

**lemma** *le-natceiling-iff*:
  **fixes** $n :: nat$ **and** $r :: real$
  **shows** $n \leq r \Longrightarrow n \leq nat(ceiling\ r)$
⟨*proof*⟩

**lemma** *natceiling-le-iff*:
  **fixes** $n :: nat$ **and** $r :: real$
  **shows** $r \leq n \Longrightarrow nat(ceiling\ r) \leq n$

⟨*proof*⟩

**lemma** *dist-real-noabs-less*:
**fixes** *a b c* :: *real* **assumes** *dist a b* < *c* **shows** *a* − *b* < *c*
⟨*proof*⟩

**lemma** *n-choose-2-nat*:
**fixes** *n* :: *nat* **shows** (*n choose 2*) = (*n* ∗ (*n* − *1*)) *div 2*
⟨*proof*⟩

**lemma** *powr-less-one*:
**fixes** *x*::*real*
**assumes** *1* < *x y* < *0*
**shows** *x powr y* < *1*
⟨*proof*⟩

**lemma** *powr-le-one-le*: ⋀*x y*::*real*. *0* < *x* ⟹ *x* ≤ *1* ⟹ *1* ≤ *y* ⟹ *x powr y* ≤ *x*
⟨*proof*⟩

## 1.2 Lists and Sets

**lemma** *list-set-tl*: *x* ∈ *set* (*tl xs*) ⟹ *x* ∈ *set xs*
⟨*proof*⟩

**lemma** *list-exhaust3*:
**obtains** *xs* = [] | *x* **where** *xs* = [*x*] | *x y ys* **where** *xs* = *x* # *y* # *ys*
⟨*proof*⟩

**lemma** *card-Ex-subset*:
*k* ≤ *card M* ⟹ ∃ *N*. *N* ⊆ *M* ∧ *card N* = *k*
⟨*proof*⟩

## 1.3 Limits and eventually

We employ filters and the *eventually* predicate to deal with the ∃ *N*. ∀ *n*≥*N*. *P n* cases. To make this more convenient, introduce a shorter syntax.

**abbreviation** *evseq* :: (*nat* ⇒ *bool*) ⇒ *bool* (**binder** ∀$^\infty$ *10*) **where**
*evseq P* ≡ *eventually P sequentially*

**lemma** *eventually-le-le*:
**fixes** *P* :: *'a* => (*'b* :: *preorder*)
**assumes** *eventually* (λ*x*. *P x* ≤ *Q x*) *net*
**assumes** *eventually* (λ*x*. *Q x* ≤ *R x*) *net*
**shows** *eventually* (λ*x*. *P x* ≤ *R x*) *net*
⟨*proof*⟩

**lemma** *LIMSEQ-neg-powr*:
**assumes** *s*: *s* < *0*
**shows** (%*x*. (*real x*) *powr s*) ⟶ *0*

⟨*proof*⟩

**lemma** *LIMSEQ-inv-powr*:
  **assumes** $0 < c$ $0 < d$
  **shows** $(\lambda n :: nat.\ (c\ /\ n)\ powr\ d) \longrightarrow 0$
⟨*proof*⟩

**end**
**theory** *Ugraphs*
**imports**
  *Girth-Chromatic-Misc*
**begin**

# 2 Undirected Simple Graphs

In this section, we define some basics of graph theory needed to formalize the Chromatic-Girth theorem.

For readability, we introduce synonyms for the types of vertexes, edges, graphs and walks.

**type-synonym** *uvert* = *nat*
**type-synonym** *uedge* = *nat set*
**type-synonym** *ugraph* = *uvert set* × *uedge set*
**type-synonym** *uwalk* = *uvert list*

**abbreviation** *uedges* :: *ugraph* ⇒ *uedge set* **where**
  *uedges G* ≡ *snd G*

**abbreviation** *uverts* :: *ugraph* ⇒ *uvert set* **where**
  *uverts G* ≡ *fst G*

**fun** *mk-uedge* :: *uvert* × *uvert* ⇒ *uedge* **where**
   *mk-uedge* (*u*,*v*) = {*u*,*v*}

All edges over a set of vertexes $S$:

**definition** *all-edges S* ≡ *mk-uedge* ‘ {*uv* ∈ *S* × *S*. *fst uv* ≠ *snd uv*}

**definition** *uwellformed* :: *ugraph* ⇒ *bool* **where**
  *uwellformed G* ≡ (∀ *e*∈*uedges G*. *card e* = 2 ∧ (∀ *u* ∈ *e*. *u* ∈ *uverts G*))

**fun** *uwalk-edges* :: *uwalk* ⇒ *uedge list* **where**
    *uwalk-edges* [] = []
  | *uwalk-edges* [*x*] = []
  | *uwalk-edges* (*x* # *y* # *ys*) = {*x*,*y*} # *uwalk-edges* (*y* # *ys*)

**definition** *uwalk-length* :: *uwalk* ⇒ *nat* **where**
  *uwalk-length p* ≡ *length* (*uwalk-edges p*)

**definition** *uwalks* :: *ugraph* ⇒ *uwalk set* **where**
*uwalks G* ≡ {*p*. *set p* ⊆ *uverts G* ∧ *set* (*uwalk-edges p*) ⊆ *uedges G* ∧ *p* ≠ []}

**definition** *ucycles* :: *ugraph* ⇒ *uwalk set* **where**
*ucycles G* ≡ {*p*. *uwalk-length p* ≥ *3* ∧ *p* ∈ *uwalks G* ∧ *distinct* (*tl p*) ∧ *hd p* = *last p*}

**definition** *remove-vertex* :: *ugraph* ⇒ *nat* ⇒ *ugraph* (- −− - [*60*,*60*] *60*) **where**
*remove-vertex G u* ≡ (*uverts G* − {*u*}, *uedges G* − {*A* ∈ *uedges G*. *u* ∈ *A*})

## 2.1 Basic Properties

**lemma** *uwalk-length-conv*: *uwalk-length p* = *length p* − *1*
⟨*proof*⟩

**lemma** *all-edges-mono*:
*vs* ⊆ *ws* ⟹ *all-edges vs* ⊆ *all-edges ws*
⟨*proof*⟩

**lemma** *all-edges-subset-Pow*: *all-edges A* ⊆ *Pow A*
⟨*proof*⟩

**lemma** *in-mk-uedge-img*: (*a*,*b*) ∈ *A* ∨ (*b*,*a*) ∈ *A* ⟹ {*a*,*b*} ∈ *mk-uedge* ‘ *A*
⟨*proof*⟩

**lemma** *distinct-edgesI*:
**assumes** *distinct p* **shows** *distinct* (*uwalk-edges p*)
⟨*proof*⟩

**lemma** *finite-ucycles*:
**assumes** *finite* (*uverts G*)
**shows** *finite* (*ucycles G*)
⟨*proof*⟩

**lemma** *ucycles-distinct-edges*:
**assumes** *c* ∈ *ucycles G* **shows** *distinct* (*uwalk-edges c*)
⟨*proof*⟩

**lemma** *card-left-less-pair*:
**fixes** *A* :: (*'a* :: *linorder*) *set*
**assumes** *finite A*
**shows** *card* {(*a*,*b*). *a* ∈ *A* ∧ *b* ∈ *A* ∧ *a* < *b*}
= (*card A* ∗ (*card A* − *1*)) *div 2*
⟨*proof*⟩

**lemma** *card-all-edges*:
**assumes** *finite A*
**shows** *card* (*all-edges A*) = *card A choose 2*

⟨*proof*⟩

**lemma** *verts-Gu*: *uverts* (*G* −− *u*) = *uverts G* − {*u*}
⟨*proof*⟩

**lemma** *edges-Gu*: *uedges* (*G* −− *u*) ⊆ *uedges G*
⟨*proof*⟩

## 2.2 Girth, Independence and Vertex Colorings

**definition** *girth* :: *ugraph* ⇒ *enat* **where**
*girth G* ≡ *INF p*∈ *ucycles G*. *enat* (*uwalk-length p*)

**definition** *independent-sets* :: *ugraph* ⇒ *uvert set set* **where**
*independent-sets Gr* ≡ {*vs*. *vs* ⊆ *uverts Gr* ∧ *all-edges vs* ∩ *uedges Gr* = {}}

**definition** α :: *ugraph* ⇒ *enat* **where**
α *G* ≡ *SUP vs* ∈ *independent-sets G*. *enat* (*card vs*)

**definition** *vertex-colorings* :: *ugraph* ⇒ *uvert set set set* **where**
*vertex-colorings G* ≡ {*C*. ⋃ *C* = *uverts G* ∧ (∀ *c1*∈*C*. ∀ *c2*∈*C*. *c1* ≠ *c2* ⟶ *c1* ∩ *c2* = {}) ∧
(∀ *c*∈*C*. *c* ≠ {} ∧ (∀ *u* ∈ *c*. ∀ *v* ∈ *c*. {*u*,*v*} ∉ *uedges G*))}

The chromatic number χ:

**definition** *chromatic-number* :: *ugraph* ⇒ *enat* **where**
*chromatic-number G* ≡ *INF c*∈ (*vertex-colorings G*). *enat* (*card c*)

**lemma** *independent-sets-mono*:
*vs* ∈ *independent-sets G* ⟹ *us* ⊆ *vs* ⟹ *us* ∈ *independent-sets G*
⟨*proof*⟩

**lemma** *le-α-iff*:
**assumes** *0* < *k*
**shows** *k* ≤ α *Gr* ⟷ *k* ∈ *card* ‘ *independent-sets Gr* (**is** *?L* ⟷ *?R*)
⟨*proof*⟩

**lemma** *zero-less-α*:
**assumes** *uverts G* ≠ {}
**shows** *0* < α *G*
⟨*proof*⟩

**lemma** α-*le-card*:
**assumes** *finite* (*uverts G*)
**shows** α *G* ≤ *card*(*uverts G*)
⟨*proof*⟩

**lemma** α-*fin*: *finite* (*uverts G*) ⟹ α *G* ≠ ∞
⟨*proof*⟩

**lemma** *α-remove-le*:
**shows** $\alpha\ (G --\ u) \leq \alpha\ G$
⟨*proof*⟩

A lower bound for the chromatic number of a graph can be given in terms of the independence number

**lemma** *chromatic-lb*:
**assumes** *wf-G*: *uwellformed G*
**and** *fin-G*: *finite* (*uverts G*)
**and** *neG*: *uverts G* ≠ {}
**shows** *card* (*uverts G*) / *α G* ≤ *chromatic-number G*
⟨*proof*⟩

**end**
**theory** *Girth-Chromatic*
**imports**
*Ugraphs*
*Girth-Chromatic-Misc*
*HOL−Probability.Probability*
*HOL−Decision-Procs.Approximation*
**begin**

# 3 Probability Space on Sets of Edges

**definition** *cylinder* :: *'a set* ⇒ *'a set* ⇒ *'a set* ⇒ *'a set set* **where**
*cylinder S A B* = {*T* ∈ *Pow S*. *A* ⊆ *T* ∧ *B* ∩ *T* = {}}

**lemma** *full-sum*:
**fixes** *p* :: *real*
**assumes** *finite S*
**shows** (∑ *A*∈*Pow S*. *p*^*card A* ∗ (*1* − *p*)^*card* (*S* − *A*)) = *1*
⟨*proof*⟩

Definition of the probability space on edges:

**locale** *edge-space* =
**fixes** *n* :: *nat* **and** *p* :: *real*
**assumes** *p-prob*: *0* ≤ *p p* ≤ *1*
**begin**

**definition** *S-verts* :: *nat set* **where**
*S-verts* ≡ {*1..n*}

**definition** *S-edges* :: *uedge set* **where**
*S-edges* = *all-edges S-verts*

**definition** *edge-ugraph* :: *uedge set* ⇒ *ugraph* **where**
*edge-ugraph es* ≡ (*S-verts*, *es* ∩ *S-edges*)

**definition** $P$ = *point-measure* (*Pow S-edges*) ($\lambda s.\ p\hat{}card\ s * (1 - p)\hat{}card\ (S\text{-}edges - s)$)

**lemma** *finite-verts*[*intro*!]: *finite S-verts*
⟨*proof*⟩

**lemma** *finite-edges*[*intro*!]: *finite S-edges*
⟨*proof*⟩

**lemma** *finite-graph*[*intro*!]: *finite* (*uverts* (*edge-ugraph es*))
⟨*proof*⟩

**lemma** *uverts-edge-ugraph*[*simp*]: *uverts* (*edge-ugraph es*) = *S-verts*
⟨*proof*⟩

**lemma** *uedges-edge-ugraph*[*simp*]: *uedges* (*edge-ugraph es*) = *es* ∩ *S-edges*
⟨*proof*⟩

**lemma** *space-eq*: *space P* = *Pow S-edges* ⟨*proof*⟩

**lemma** *sets-eq*: *sets P* = *Pow* (*Pow S-edges*) ⟨*proof*⟩

**lemma** *emeasure-eq*:
*emeasure P A* = (*if A* ⊆ *Pow S-edges then* ($\sum$ *edges*∈*A*. *p*^*card edges* * (1 − *p*)^*card* (*S-edges* − *edges*)) *else 0*)
⟨*proof*⟩

**lemma** *integrable-P*[*intro*, *simp*]: *integrable P* (*f*::- ⇒ *real*)
⟨*proof*⟩

**lemma** *borel-measurable-P*[*measurable*]: *f* ∈ *borel-measurable P*
⟨*proof*⟩

**lemma** *prob-space-P*: *prob-space P*
⟨*proof*⟩

**end**

**sublocale** *edge-space* ⊆ *prob-space P*
⟨*proof*⟩

**context** *edge-space*
**begin**

**lemma** *prob-eq*:
*prob A* = (*if A* ⊆ *Pow S-edges then* ($\sum$ *edges*∈*A*. *p*^*card edges* * (1 − *p*)^*card* (*S-edges* − *edges*)) *else 0*)
⟨*proof*⟩

**lemma** *integral-finite-singleton*: $integral^L\ P\ f = (\sum x{\in}Pow\ S\text{-}edges.\ f\ x * measure\ P\ \{x\})$
⟨*proof*⟩

Probability of cylinder sets:

**lemma** *cylinder-prob*:
**assumes** $A \subseteq S\text{-}edges\ B \subseteq S\text{-}edges\ A \cap B = \{\}$
**shows** $prob\ (cylinder\ S\text{-}edges\ A\ B) = p\ \hat{}\ (card\ A) * (1 - p)\ \hat{}\ (card\ B)$ (**is** - = *?pp A B*)
⟨*proof*⟩

**lemma** *Markov-inequality*:
**fixes** $a :: real$ **and** $X :: uedge\ set \Rightarrow real$
**assumes** $0 < c\ \bigwedge x.\ 0 \leq f\ x$
**shows** $prob\ \{x \in space\ P.\ c \leq f\ x\} \leq (\int x.\ f\ x\ \partial\ P)\ /\ c$
⟨*proof*⟩

**end**

### 3.1 Graph Probabilities outside of *Edge-Space* locale

These abbreviations allow a compact expression of probabilities about random graphs outside of the *Edge-Space* locale. We also transfer a few of the lemmas we need from the locale into the toplevel theory.

**abbreviation** $MGn :: (nat \Rightarrow real) \Rightarrow nat \Rightarrow (uedge\ set)\ measure$ **where**
$MGn\ p\ n \equiv (edge\text{-}space.P\ n\ (p\ n))$
**abbreviation** $probGn :: (nat \Rightarrow real) \Rightarrow nat \Rightarrow (uedge\ set \Rightarrow bool) \Rightarrow real$ **where**
$probGn\ p\ n\ P \equiv measure\ (MGn\ p\ n)\ \{es \in space\ (MGn\ p\ n).\ P\ es\}$

**lemma** *probGn-le*:
**assumes** *p-prob*: $0 < p\ n\ p\ n < 1$
**assumes** *sub*: $\bigwedge n\ es.\ es \in space\ (MGn\ p\ n) \Longrightarrow P\ n\ es \Longrightarrow Q\ n\ es$
**shows** $probGn\ p\ n\ (P\ n) \leq probGn\ p\ n\ (Q\ n)$
⟨*proof*⟩

## 4 Short cycles

**definition** *short-cycles* :: *ugraph* ⇒ *nat* ⇒ *uwalk set* **where**
*short-cycles G k* ≡ {*p* ∈ *ucycles G*. *uwalk-length p* ≤ *k*}

obtains a vertex in a short cycle:

**definition** *choose-v* :: *ugraph* ⇒ *nat* ⇒ *uvert* **where**
*choose-v G k* ≡ *SOME u*. ∃*p*. *p* ∈ *short-cycles G k* ∧ *u* ∈ *set p*

**partial-function** (*tailrec*) *kill-short* :: *ugraph* ⇒ *nat* ⇒ *ugraph* **where**
*kill-short G k* = (*if short-cycles G k* = {} *then G else* (*kill-short* (*G* −− (*choose-v G k*)) *k*))

**lemma** *ksc-simps*[*simp*]:
*short-cycles G k* = {} ⟹ *kill-short G k* = *G*
*short-cycles G k* ≠ {} ⟹ *kill-short G k* = *kill-short* (*G* −− (*choose-v G k*)) *k*
⟨*proof*⟩

**lemma**
**assumes** *short-cycles G k* ≠ {}
**shows** *choose-v--in-uverts*: *choose-v G k* ∈ *uverts G* (**is** *?t1*)
**and** *choose-v--in-short*: ∃ *p*. *p* ∈ *short-cycles G k* ∧ *choose-v G k* ∈ *set p* (**is** *?t2*)
⟨*proof*⟩

**lemma** *kill-step-smaller*:
**assumes** *short-cycles G k* ≠ {}
**shows** *short-cycles* (*G* −− (*choose-v G k*)) *k* ⊂ *short-cycles G k*
⟨*proof*⟩

Induction rule for *kill-short*:

**lemma** *kill-short-induct*[*consumes 1*, *case-names empty kill-vert*]:
**assumes** *fin*: *finite* (*uverts G*)
**assumes** *a-empty*: ⋀*G*. *short-cycles G k* = {} ⟹ *P G k*
**assumes** *a-kill*: ⋀*G*. *finite* (*short-cycles G k*) ⟹ *short-cycles G k* ≠ {}
⟹ *P* (*G* −− (*choose-v G k*)) *k* ⟹ *P G k*
**shows** *P G k*
⟨*proof*⟩

Large Girth (after *kill-short*):

**lemma** *kill-short-large-girth*:
**assumes** *finite* (*uverts G*)
**shows** *k* < *girth* (*kill-short G k*)
⟨*proof*⟩

Order of graph (after *kill-short*):

**lemma** *kill-short-order-of-graph*:
**assumes** *finite* (*uverts G*)
**shows** *card* (*uverts G*) − *card* (*short-cycles G k*) ≤ *card* (*uverts* (*kill-short G k*))
⟨*proof*⟩

Independence number (after *kill-short*):

**lemma** *kill-short-α*:
**assumes** *finite* (*uverts G*)
**shows** *α* (*kill-short G k*) ≤ *α G*
⟨*proof*⟩

Wellformedness (after *kill-short*):

**lemma** *kill-short-uwellformed*:
**assumes** *finite* (*uverts G*) *uwellformed G*

**shows** *uwellformed* (*kill-short G k*)
⟨*proof*⟩

# 5 The Chromatic-Girth Theorem

Probability of Independent Edges:

**lemma** (**in** *edge-space*) *random-prob-independent*:
**assumes** $n \geq k\ k \geq 2$
**shows** *prob* $\{es \in space\ P.\ k \leq \alpha\ (edge\text{-}ugraph\ es)\}$
$\leq (n\ choose\ k) * (1-p) \hat{}\ (k\ choose\ 2)$
⟨*proof*⟩

Almost never many independent edges:

**lemma** *almost-never-le-α*:
**fixes** $k :: nat$
**and** $p :: nat \Rightarrow real$
**assumes** *p-prob*: $\forall^{\infty}\ n.\ 0 < p\ n \wedge p\ n < 1$
**assumes** [*arith*]: $k > 0$
**assumes** *N-prop*: $\forall^{\infty}\ n.\ (6 * k * ln\ n)/n \leq p\ n$
**shows** $(\lambda n.\ probGn\ p\ n\ (\lambda es.\ 1/2*n/k \leq \alpha\ (edge\text{-}space.edge\text{-}ugraph\ n\ es))) \longrightarrow 0$
(**is** $(\lambda n.\ ?prob\text{-}fun\ n) \longrightarrow 0$)
⟨*proof*⟩

Mean number of k-cycles in a graph. (Or rather of paths describing a circle of length $k$):

**lemma** (**in** *edge-space*) *mean-k-cycles*:
**assumes** $3 \leq k\ k < n$
**shows** $(\int es.\ card\ \{c \in ucycles\ (edge\text{-}ugraph\ es).\ uwalk\text{-}length\ c = k\}\ \partial\ P)$
$= of\text{-}nat\ (fact\ n\ div\ fact\ (n-k)) * p\ \hat{}\ k$
⟨*proof*⟩

Girth-Chromatic number theorem:

**theorem** *girth-chromatic*:
**fixes** $l :: nat$
**shows** $\exists\ G.\ uwellformed\ G \wedge l < girth\ G \wedge l < chromatic\text{-}number\ G$
⟨*proof*⟩

**end**

# References


[1] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, 4 edition, 2010. http://diestel-graph-theory.com.