goals accomplished 🎉

we formalized Hall’s Marriage Theorem

Alena Gusakov
Joint work with Kyle Miller and Bhavik Mehta

arXiv:2101.00127

January 7th - Lean Together 2021

What is this?

Hall’s Marriage Theorem is a standard part of the undergraduate discrete mathematics curriculum.

It’s not in mathlib yet.

We’re working on it.

Outline

- Adopting dogs
- Three ways of formulating the theorem
- The proof
- Three ways of formalizing the theorem
- The Lean proof

Adopting dogs

Three Ways of Formulating

- Indexed families of finite sets
- Relations between types
- Matchings in bipartite graphs

Indexed families of finite sets

Definition 2.1.1. For a fixed set S, a *family of finite subsets* $\{X_i\}_{i\in I}$ *indexed by a set* I is a collection of subsets $X_i \subseteq S$ for each $i \in I$. The set I is called the *index set*. An element $x \in \prod_{i\in I} X_i$ is called a *family of elements* of the indexed family, and it may be regarded as a function $I \to S$ with $x_i := x(i)$ with $x_i \in X_i$ for each $i \in I$.

Definition 2.1.2. A *matching* (or *transversal*) of an indexed family of subsets $\{x_i\}_{i\in I}$ is a family of elements x that is injective when thought of as a function $I \to S$, which is to say that $x_i = x_j$ implies $i = j$.

Theorem 2.1.3 (Hall's Marriage Theorem [Hal35]). *Let* $\{X_i\}_{i\in I}$ *be an indexed family of finite subsets with finite index set* I. *The indexed family has a matching if and only if for all* $J \subseteq I$, *we have* $|J| \leq \left|\bigcup_{i\in J} X_i\right|$.

Relations between types

For sets A and B, consider a relation r between A and B, with $r\ a\ b$ indicating that $a \in A$ is related to $b \in B$ by r. For a subset $S \subseteq A$, let $r(S)$ denote the set $\{b \in B \mid \exists a \in A, r\ a\ b\}$.

Definition 2.2.1. Given a relation r between sets A and B, a *matching of r that saturates a subset $S \subseteq A$* is an injective function $f : S \to B$ that *respects* the relation r, which is to say that $r\ a\ f(a)$ for all $a \in S$. A matching that saturates A is simply called a matching.

Theorem 2.2.2 (Hall's Marriage Theorem). *Let r be a relation between a finite set A and a finite set B. The relation has a matching that saturates A if and only if for all $S \subseteq A$ then $|S| \leq |r(S)|$.*

Matchings in bipartite graphs

A *(simple) graph* G on a set V of *vertices* is a symmetric irreflexive binary relation on V, where vertices $v, w \in V$ are *adjacent* if they are related by this relation. An *edge* of G is an unordered pair of adjacent vertices, and the set of all edges of G is denoted $E(G)$; the vertices comprising an edge are said to be *incident* to it. For subsets $S \subseteq V$ of vertices, the *neighborhood* $\Gamma(S)$ of S is the set of all vertices in V adjacent to at least one vertex in S.

Definition 2.3.1. A *matching* M on a graph G is a subset $M \subseteq E(G)$ of edges such that distinct edges of M share no incident vertices. The matching is said to *saturate* a subset $W \subseteq V$ if every vertex of W is incident to an edge of M.

Definition 2.3.2. A *(proper) coloring* of a graph G with color set C is a function $f : V \to C$ assigning colors to each vertex such that adjacent vertices have different colors. For color $c \in C$, the *color class* associated to c is $f^{-1}(c)$.

Definition 2.3.3. A *bipartition* of a graph G is a coloring of G with color set $\{1, 2\}$. Let V_1 and V_2 respectively denote the color classes for colors 1 and 2. If a bipartition exists, the graph is called *bipartite*.

Theorem 2.3.4 (Hall's Marriage Theorem). *Let G be a bipartitioned simple graph with V_1 finite and $\Gamma(v)$ finite for each $v \in V_1$. G has a matching that saturates V_1 if and only if for all $S \subseteq V_1$ then $|S| \leq |\Gamma(S)|$.*

Proof

Theorem 2.2.2 (Hall's Marriage Theorem). *Let r be a relation between a finite set A and a finite set B. The relation has a matching that saturates A if and only if for all $S \subseteq A$ then $|S| \leq |r(S)|$.*

Proof. First suppose that there exists a matching M that saturates A. If $S \subseteq A$, then since M saturates A it must also saturate S. If $M(S)$ denotes the image of S by M in B, then $|S| = |M(S)|$ by injectivity. Since $M(S) \subseteq r(S)$, we have that $|S| = |M(S)| \leq |r(S)|$.

The converse is the "hard" direction. We proceed by strong induction on $n = |A|$.

Base case ($n = 0$): This means that $A = \emptyset$. The empty matching saturates $\emptyset$.

Base case ($n = 1$): This means that $A = \{a\}$ for some a, hence every $S \subseteq A$ is either the empty set or $\{a\}$. Since we have that $|S| \leq |r(S)|$ for every $S \subseteq A$, we know that $|\{a\}| \leq |r(\{a\})|$, so there exists some $b \in B$ such that $r\ a\ b$. We can define our matching as the function $f : A \to B$ such that $f(a) = b$.

Induction hypothesis: If r is a relation between a finite set A with $|A| \leq k$ and a finite set B, then if $|S| \leq |r(S)|$ for every $S \subseteq A$, there exists a matching of r that saturates A.

Induction step: Suppose $|A| = k + 1$ and $|S| \leq |r(S)|$ for every $S \subseteq A$. We have two cases: either (1) every proper nonempty subset $S \subsetneq A$ satisfies $|S| < |r(S)|$ or (2) there is some proper nonempty subset $S \subsetneq A$ such that $|S| = |r(S)|$.

Case 1: Assume for every nonempty subset $S \subsetneq A$ that $|S| < |r(S)|$, and choose arbitrary $a \in A$ and $b \in r(\{a\})$. Set $A' := A \setminus \{a\}$ and $B' := B \setminus \{b\}$, and let r' be the restriction of r to A' and B'. We prove that Hall's condition is satisfied for r'. Let $T \subseteq A'$. Since $|T| < |r(T)|$, we know that $|T| + 1 \leq |r(T)|$, and removing b from B gives us $|r(T)| - 1 \leq |r'(T)|$, so we now have that $|T| \leq |r'(T)|$. By our induction hypothesis, there exists a matching $M' : A' \to B'$, which can be extended to a matching $M : A \to B$ with $M(a) = b$.

Case 2: There exists some proper nonempty $S_0 \subsetneq A$ such that $|S_0| = |r(S_0)|$. We first prove that Hall's condition is satisfied for S_0. We restrict r to a relation r' between S_0 and $r(S_0)$, hence for $T \subseteq S_0$ we have $r(T) = r'(T)$. Since for all $T \subseteq S_0$, $|S_0| \leq k$ and $|T| = |r'(T)|$, by our induction hypothesis there is a matching M_0 of r' that saturates S_0.

Now we consider $A'' = A \setminus S_0$ and $B'' = B \setminus r(S_0)$. Let r'' be the restriction of r to A'' and B''. Thus, for $T \subseteq A'$,

$$r''(T) = \{y \mid r\ x\ y \text{ for some } x \in T \text{ and } y \in B'\}.$$

Since T and S_0 are disjoint and $r''(T)$ and $r'(S_0)$ are disjoint, we have that $|S_0 \cup T| = |S_0| + |T|$, and $r(S_0 \cup T) = r'(S_0) \cup r''(T)$ so therefore $|r(S_0 \cup T)| = |r'(S_0)| + |r''(T)|$. Since $|S| \leq |r(S)|$ for all $S \subseteq A$, we have that $|S_0| + |T| = |S_0 \cup T| \leq |r(S_0 \cup T)| = |r'(S_0)| + |r''(T)|$, so $|S_0| + |T| \leq |r'(S_0)| + |r''(T)|$. Since $|S_0| = |r'(S_0)|$, we therefore have $|T| \leq |r''(T)|$ for all $T \subset A''$. By our induction hypothesis, this means we have a matching M_1 for r'' that saturates A''.

Since the domains of M_0 and M_1 are disjoint, we can define a matching M that saturates A by $M(a) = M_0(a)$ for $a \in S_0$ and $M(a) = M_1(a)$ otherwise.

This completes the proof. $\square$

Three Ways of Formalizing

- Indexed families of finite sets
- Relations between types
- Matchings in bipartite graphs

Indexed families of finite sets

```
universes u v
variables {α : Type u} {β : Type v} (ι : α → finset β)

structure matching :=
(f : α → β)
(mem_prod' : ∀ (a : α), f a ∈ ι a)
(injective' : injective f)


theorem hall [fintype α] :
  (∀ (s : finset α), s.card ≤ (s.bind ι).card) ↔ nonempty (matching ι)
```

Relations between types

```
variables {α β : Type u} [fintype α] [fintype β]
variables (r : α → β → Prop)
def image_rel (A : finset α) : finset β := univ.filter (λ b, ∃ a ∈ A, r a b)
```

```
theorem hall :
  (∀ (A : finset α), A.card ≤ (image_rel r A).card)
    ↔ (∃ (f : α → β), function.injective f ∧ ∀ x, r x (f x))
```

Matchings in bipartite graphs - simple graphs

```
structure simple_graph (V : Type u) :=
(adj : V → V → Prop)
(sym : symmetric adj)
(loopless : irreflexive adj)

/-- The set of all `w` adjacent to a given `v`. -/
def neighbor_set (v : V) : set V := {w : V | G.adj v w}

/-- The set of all `w` adjacent to an element of `S`. -/
def neighbor_set_image (S : set V) : set V :=
{w : V | ∃ v, v ∈ S ∧ w ∈ G.neighbor_set v}

/-- The set of all unordered pairs `⟦(v, w)⟧` such that `G.adj v w` -/
def edge_set : set (sym2 V) := sym2.from_rel G.sym
```

Matchings in bipartite graphs - bipartitions

```
/-- `G.coloring C` is the type of `C`-colorings of `G`. -/
structure coloring (G : simple_graph V) (C : Type v) :=
(color : V → C)
-- Adjacent vertices have distinct colors:
(valid : ∀ ⦃v w : V⦄, G.adj v w → color v ≠ color w)

/-- The set of vertices in the color class for `c`. -/
def coloring.color_set (c : C) : set V := f.color ⁻¹' {c}

/-- A bipartition `f : G.bipartition` is a coloring of `G` by
    the two-term type `fin 2`.  The color classes `f.color_set 0`
    and `f.color_set 1` give the partition of `V`. -/
def bipartition (G : simple_graph V) := G.coloring (fin 2)
```

Matchings in bipartite graphs - theorem

```
structure matching (G : simple_graph V) :=
(edges : set (sym2 V))
(sub_edges : edges ⊆ G.edge_set)
-- If two edges are in the matching, and if v is a vertex incident to both,
-- then the edges are the same:
(disjoint : ∀ (x y ∈ edges) (v : V), v ∈ x → v ∈ y → x = y)

def matching.saturates (M : G.matching) (S : set V) : Prop :=
S ⊆ {v : V | ∃ x, x ∈ M.edges ∧ v ∈ x}
```

```
variables (G : simple_graph V) [fintype V] (b : G.bipartition)

theorem hall_marriage_theorem :
  (∀ (S ⊆ (b.color_set 0)),
     fintype.card S ≤ fintype.card (G.neighbor_set_image S))
  ↔ (∃ (M : G.matching), M.saturates (b.color_set 0))
```

Formalized Proof - Easy direction & base cases

```
variables {α β : Type u} [fintype α] [fintype β]
variables (r : α → β → Prop)
def image_rel (A : finset α) : finset β := univ.filter (λ b, ∃ a ∈ A, r a b)

theorem hall :
  (∀ (A : finset α), A.card ≤ (image_rel r A).card)
    ↔ (∃ (f : α → β), function.injective f ∧ ∀ x, r x (f x))

theorem hall_easy (f : α → β) (hf₁ : function.injective f) (hf₂ : ∀ x, r x (f x))
(A : finset α) : A.card ≤ (image_rel r A).card

theorem hall_hard_inductive_zero (hn : fintype.card α = 0)
  (hr : ∀ (A : finset α), A.card ≤ (image_rel r A).card) :
  ∃ (f : α → β), function.injective f ∧ ∀ x, r x (f x)

theorem hall_hard_inductive_one (hn : fintype.card α = 1)
  (hr : ∀ (A : finset α), A.card ≤ (image_rel r A).card) :
  ∃ (f : α → β), function.injective f ∧ ∀ x, r x (f x)
```

Formalized Proof - Hard direction induction

```
lemma hall_hard_inductive_step_1 [nontrivial α] {n : ℕ}
  (hn : fintype.card α ≤ n.succ)
  (ha : ∀ (A : finset α), A.nonempty → A ≠ univ → A.card < (image_rel r A).card)
  (ih : ∀ {α' β' : Type u} [fintype α'] [fintype β'] (r' : α' → β' → Prop),
    fintype.card α' ≤ n →
    (∀ (A' : finset α'), A'.card ≤ (image_rel r' A').card) →
    ∃ (f' : α' → β'), function.injective f' ∧ ∀ x, r' x (f' x)) :
  ∃ (f : α → β), function.injective f ∧ ∀ x, r x (f x)
```

```
lemma hall_hard_inductive_step_2 [nontrivial α] {n : ℕ}
  (hn : fintype.card α ≤ n.succ)
  (hr : ∀ (A : finset α), A.card ≤ (image_rel r A).card)
  (ha : ∃ (A : finset α),  A.nonempty ∧ A ≠ univ ∧ A.card = (image_rel r A).card)
  (ih : ∀ {α' β' : Type u} [fintype α'] [fintype β'] (r' : α' → β' → Prop),
    fintype.card α' ≤ n →
    (∀ (A' : finset α'), A'.card ≤ (image_rel r' A').card) →
    ∃ (f' : α' → β'), function.injective f' ∧ ∀ x, r' x (f' x)) :
  ∃ (f : α → β), function.injective f ∧ ∀ x, r x (f x)
```

Next Steps

We have the countably infinite Hall

```
theorem infinite_hall {α : Type u} {β : Type v} (ι : α → finset β) (h : ℕ ≃ α) :
  (∀ (s : finset α), s.card ≤ (s.bind ι).card) ↔ nonempty (matching ι)
```

Next Steps

We have the countably infinite Hall

We want to use the category theory library to prove the full infinite Hall

```
theorem infinite_hall {α : Type u} {β : Type v} (ι : α → finset β) (h : ℕ ≃ α) :
  (∀ (s : finset α), s.card ≤ (s.bind ι).card) ↔ nonempty (matching ι)
```

$\cancel{(h : \mathbb{N} \simeq \alpha)}$

Thanks 🎉

For more details, see arXiv:2101.00127