Mechanizing the Odd Order Theorem: Local Analysis

In honour of Gérard Berry and Jean-Jacques Lévy

Assia Mahboubi

INRIA Microsoft Research Joint Centre (France)
INRIA Saclay – Île-de-France
École Polytechnique, Palaiseau

February 8, 2011

Finite group theory

Groups are algebraic structures closed under an associative, invertible law.

Finite group theory

- Elements of a group combine thanks to the group law:

$$g \quad g^{-1} \quad e \quad g * h \quad g * h * g^{-1}$$

Finite group theory

- Elements of a group combine thanks to the group law:

$$g \quad g^{-1} \quad e \quad g * h \quad g * h * g^{-1}$$

- But groups themselves combine through various kinds of operators:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

Finite group theory

- Elements of a group combine thanks to the group law:

$$g \quad g^{-1} \quad e \quad g * h \quad g * h * g^{-1}$$

- But groups themselves combine through various kinds of operators:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

- And most of the theory is developed forgetting about the points.

Decomposition

Theorem (Existence of prime decomposition)

A number always admits a decomposition into a product of prime numbers.

Theorem (Existence of composition series)

For any finite group G, there exists a sequence of subgroups:

$$\{1\} = G_0 \lhd G_1 \lhd \cdots \lhd G = G_n$$

such that for all k, G_{k+1}/G_k is simple.

Decomposition

Theorem (Existence of prime decomposition)

A number always admits a decomposition into a product of prime numbers.

Theorem (Existence of composition series)

For any finite group G, there exists a sequence of subgroups:

$$\{1\} = G_0 \lhd G_1 \lhd \cdots \lhd G = G_n$$

such that for all k, G_{k+1}/G_k is simple.

Such a sequence is called a composition series for G.
Such a quotient G_{k+1}/G_k is called a factor.

Uniqueness

Theorem (Prime decomposition uniqueness)

The decomposition of any number into a product of primes is unique up to permutations.

Theorem (Jordan-Hölder uniqueness)

For any group G, two composition series for G have the same length, and the same factors up to isomorphism and permutation.

Proving Jordan Hölder Theorem

By induction on the cardinal of G:

Proving Jordan Hölder Theorem

By induction on the cardinal of G:

- Base case: $\sharp G = 0$.
  - trivial (a group has at least one element: the neutral)

Proving Jordan Hölder Theorem

By induction on the cardinal of G:

- Base case: $\sharp G = 0$.
  - trivial (a group has at least one element: the neutral)
- Inductive case: $\sharp G > 0$:
  - If G has an empty series: then it is trivial and all its series are empty.
  - If G is simple: then all its series are trivial.
  - Else let (N_i) and (M_j) be two (non empty) composition series of G.

Jordan Hölder Theorem

By induction on the cardinal of G:

Jordan Hölder Theorem

By induction on the cardinal of G:

$$G \triangleright N_1 \triangleright N_2 \triangleright \cdots \triangleright N_r = 1$$

$$I = N_1 \cap M_1$$

$$G \triangleright M_1 \triangleright M_2 \triangleright \cdots \triangleright M_s = 1$$

Jordan Hölder Theorem

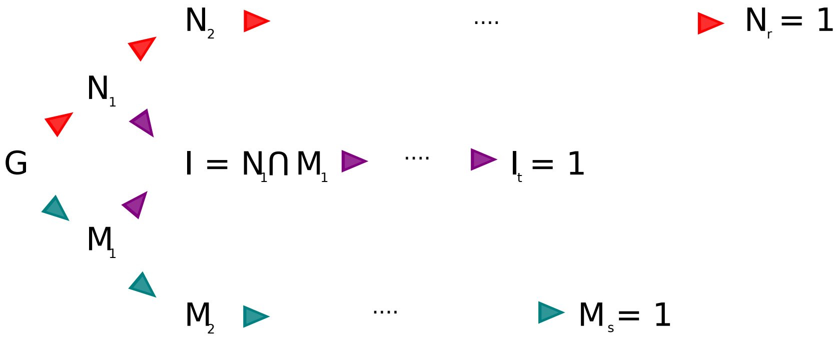
By induction on the cardinal of G:

Jordan Hölder Theorem

By induction on the cardinal of G:

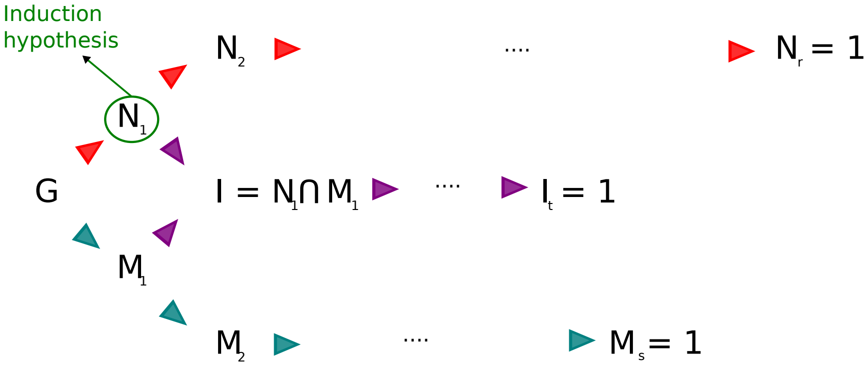
Jordan Hölder Theorem

By induction on the cardinal of G:



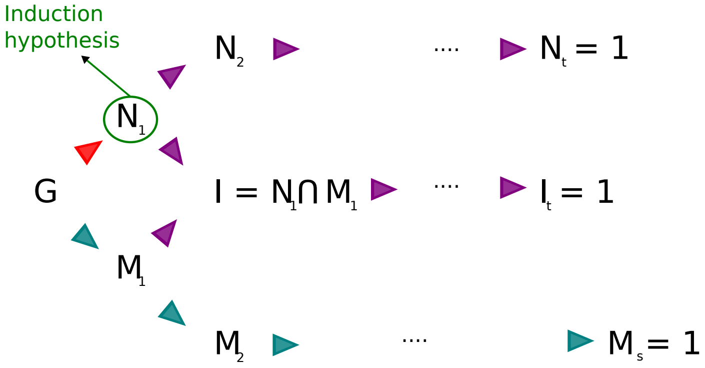
Jordan Hölder Theorem

By induction on the cardinal of G:



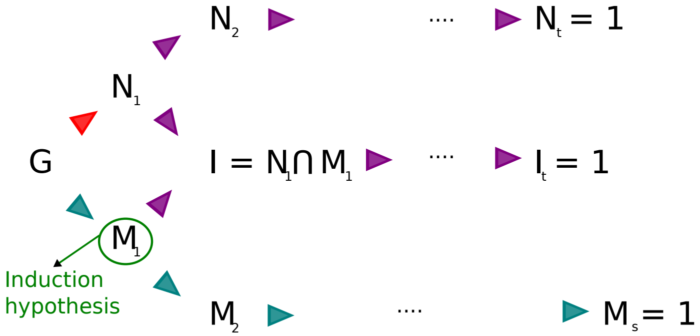
Jordan Hölder Theorem

By induction on the cardinal of G:



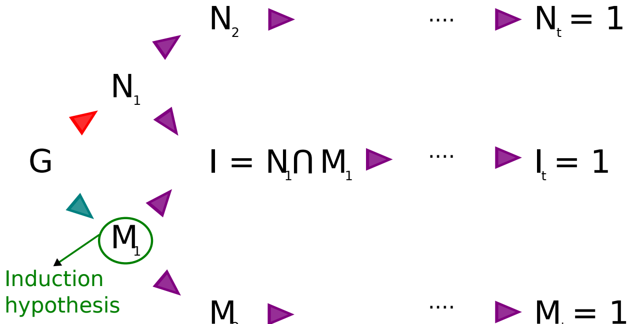
Jordan Hölder Theorem

By induction on the cardinal of G:



Jordan Hölder Theorem

By induction on the cardinal of G:



Jordan Hölder Theorem

We have obtained:

$$
\begin{tikzcd}
 & & N_2 \arrow[r, phantom, "\triangleright"] & \cdots \arrow[r, phantom, "\triangleright"] & N_t = 1 \\
 & N_1 \arrow[ur, phantom, "\triangleright" sloped] \arrow[dr, phantom, "\triangleright" sloped] & & & \\
G \arrow[ur, phantom, "\triangleright" sloped] \arrow[dr, phantom, "\triangleright" sloped] & & I = N_1 \cap M_1 \arrow[r, phantom, "\triangleright"] & \cdots \arrow[r, phantom, "\triangleright"] & I_t = 1 \\
 & M_1 \arrow[ur, phantom, "\triangleright" sloped] \arrow[dr, phantom, "\triangleright" sloped] & & & \\
 & & M_2 \arrow[r, phantom, "\triangleright"] & \cdots \arrow[r, phantom, "\triangleright"] & M_t = 1
\end{tikzcd}
$$

Jordan Hölder Theorem

We have obtained:

$$G \triangleright N_1 \triangleright N_2 \triangleright \cdots \triangleright N_t = 1$$

$$G \triangleright M_1$$

$$N_1 \triangleright I = N_1 \cap M_1 \triangleright \cdots \triangleright I_t = 1$$

$$M_1 \triangleright I$$

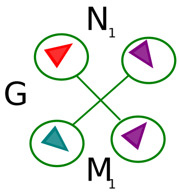
Jordan Hölder Theorem

We have obtained:

$$
\begin{array}{ccccccc}
 & N_1 & & & & & \\
G & & I = N_1 \cap M_1 & \triangleright & \cdots & \triangleright & I_t = 1 \\
 & M_1 & & & & &
\end{array}
$$

Jordan Hölder Theorem

We conclude by a "butterfly lemma":



$$I = N_1 \cap M_1 \triangleright \cdots \triangleright I_t = 1$$

Finite group decomposition (continued)

Unfortunately the analogy with arithmetics soon breaks down:

Finite group decomposition (continued)

Unfortunately the analogy with arithmetics soon breaks down:

- Two numbers with the same multiset of prime factors are equal.

Finite group decomposition (continued)

Unfortunately the analogy with arithmetics soon breaks down:

- Two numbers with the same multiset of prime factors are equal.
- Two groups with the same (up to isomorphism) multiset of prime factors are not necessarily equal (not even isomorphic).

Finite group decomposition (continued)

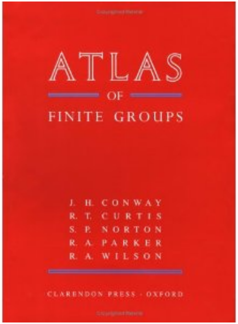
Unfortunately the analogy with arithmetics soon breaks down:

- Two numbers with the same multiset of prime factors are equal.
- Two groups with the same (up to isomorphism) multiset of prime factors are not necessarily equal (not even isomorphic).

$\Rightarrow$ Classifying finite groups is much more difficult than "classifying" numbers.

The Atlas of Finite Groups

The classification of all simple finite groups
aka. The Enormous Theorem



has been considered achieved in 1983 and revised in 2005.

The Odd Order Theorem

Theorem (Feit - Thompson (1963)) :

Every simple group of odd order is solvable.
otherwise said
Every simple group of odd order is cyclic.

The Odd Order Theorem

Theorem (Feit - Thompson (1963)) :

Every simple group of odd order is solvable.

otherwise said

Every simple group of odd order is cyclic.

- Original published proof: one entire volume of the Pacific Journal of Mathematics

The Odd Order Theorem

Theorem (Feit - Thompson (1963)) :

Every simple group of odd order is solvable.

otherwise said

Every simple group of odd order is cyclic.

- Original published proof: one entire volume of the Pacific Journal of Mathematics
- A collective simplification work: two entire volumes of London Math. Society Lecture Notes.

The Odd Order Theorem

Theorem (Feit - Thompson (1963)) :

Every simple group of odd order is solvable.
otherwise said
Every simple group of odd order is cyclic.

- Original published proof: one entire volume of the Pacific Journal of Mathematics
- A collective simplification work: two entire volumes of London Math. Society Lecture Notes.
- (Wikipedia 01/02/2011) "It takes a professional group theorist about a year of hard work to understand the proof completely."

The Odd Order Theorem

Theorem (Feit - Thompson (1963)) :

Every simple group of odd order is solvable.
otherwise said
Every simple group of odd order is cyclic.

- Original published proof: one entire volume of the Pacific Journal of Mathematics
- A collective simplification work: two entire volumes of London Math. Society Lecture Notes.
- (Wikipedia 01/02/2011) "It takes a professional group theorist about a year of hard work to understand the proof completely."
- The proof mixes many theories, non only combinatorics but also linear algebra, Galois theory, characters,...

Formalization issues: sets vs. types

Not all distinct sets should be modelled as distinct types

- A type fixes common requirements for its inhabitants.
- Type inhabitants are objects we want to observe and combine.

Formalization issues: sets vs. types

Types should have the appropriate granularity.

Formalization issues: sets vs. types

Types should have the appropriate granularity.

Remember the main objects we manipulate most are groups, not elements:

Formalization issues: sets vs. types

Types should have the appropriate granularity.
Remember the main objects we manipulate most are groups, not elements:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

Formalization issues: sets vs. types

Types should have the appropriate granularity.
Remember the main objects we manipulate most are groups, not elements:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

- Work within a finite group domain type which fixes the law.

```
gT : finGroupType
```

Formalization issues: sets vs. types

Types should have the appropriate granularity.
Remember the main objects we manipulate most are groups, not elements:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

- Work within a finite group domain type which fixes the law.
  `gT : finGroupType`
- Groups are collections (subsets) of inhabitants of this big group.
  `G H : {group gT}`

Formalization issues: sets vs. types

Types should have the appropriate granularity.
Remember the main objects we manipulate most are groups, not elements:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

- Work within a finite group domain type which fixes the law.

  `gT : finGroupType`

- Groups are collections (subsets) of inhabitants of this big group.

  `G H : {group gT}`

- Operations on groups remain homogeneous.

Formalization issues: sets vs. types

Types should have the appropriate granularity.
Remember the main objects we manipulate most are groups, not elements:

$$G \times H \quad G * H \quad G \cap H \quad G \rtimes H \quad G/H$$

- Work within a finite group domain type which fixes the law.

```
gT : finGroupType
```

- Groups are collections (subsets) of inhabitants of this big group.

```
G H : {group gT}
```

- Operations on groups remain homogeneous.
- Only change type when a new group law is really needed

quotient groups...

Formalisation issues: notations

Let $M \in M_n(F)$,

$$\det(M) := \sum_{s \in S_n} (-1)^{\epsilon_s} \prod_i M_{i,s(i)}$$

Formalisation issues: notations

Let $M \in M_n(F)$,

$$\det(M) := \sum_{s \in S_n} (-1)^{\epsilon_s} \prod_i M_{i,s(i)}$$

In LaTeX:

```
\det(M) :=
  \sum_{s \in S_n}(-1)^{\epsilon_s}\prod_i M_{i, s(i)}
```

Formalisation issues: notations

Let $M \in M_n(F)$,

$$\det(M) := \sum_{s \in S_n} (-1)^{\epsilon_s} \prod_i M_{i,s(i)}$$

In LaTeX:

```
\det(M) :=
  \sum_{s \in S_n}(-1)^{\epsilon_s}\prod_i M_{i, s(i)}
```

In a proof assistant, one would like to write:

```
Definition determinant n (A : 'M[R]_n) : R :=
  \sum_(s : 'S_n) (-1) ^+ s * \prod_i A i (s i).
```

Formalisation issues: notations

$$\sum \quad \prod \quad \bigcup \quad \bigcap \quad \oplus \ldots$$

This requires:

- Concise and uniform notations: $\displaystyle\sum_{i=0}^{n} \quad \bigcup_{A \notin E} \quad \bigcap_{A|P(A)} \ldots$
- Generic toolkit: $\displaystyle\bigcap_{A \in E} = \bigcap_{A \in E \cap B} \cap \bigcap_{A \in E \cap B^c} \ldots$
- Implicit properties of associated operators:

$$\prod_{i=1}^{n} \sum_{j=1}^{m} = \sum \prod$$

This is possible using:

- A generic operator to program these expressions;
- Higher order type inference (à la type classes);
- Coq notation mechanism.

Formalization issues : a page in finite group theory

Finite groups, vol 2. VIII.5.9 - Huppert Blackburn (excerpt):

Since $\mathfrak{A}/\mathfrak{U} = (\mathfrak{K}/\mathfrak{U})(\mathfrak{B}/\mathfrak{U})$, it follows that:

$$\mathfrak{A} = \mathfrak{K}\mathfrak{B} = \mathfrak{D}\mathfrak{U}\mathfrak{B} = \mathfrak{D}\mathfrak{B}$$

Also $\mathfrak{K} \cap \mathfrak{B} = \mathfrak{U}$ and $\mathfrak{U}\mho^p(\mathfrak{K}) \cap \mathfrak{D} = \mho^p(\mathfrak{K})$, so

$$\mathfrak{D} \cap \mathfrak{B} = \mathfrak{D} \cap \mathfrak{K} \cap \mathfrak{B} = \mathfrak{D} \cap \mathfrak{U} = \mathfrak{D} \cap \mathfrak{U}\mho^p(\mathfrak{K}) \cap \mathfrak{U} = \mho^p(\mathfrak{K}) \cap \mathfrak{U} = 1$$

Thus $\mathfrak{A} = \mathfrak{D} \times \mathfrak{B}$. Then $\mathfrak{K} = \mathfrak{D} \times (\mathfrak{B} \times \mathfrak{K}) = \mathfrak{D} \times \mathfrak{U}$, so $\mathfrak{D} \simeq \mathfrak{K}/\mathfrak{U}$ is homocylic and $\mathfrak{D}/\Phi(\mathfrak{D})$ is $\mathfrak{X}$-irreducible. Also $\mathfrak{D} \neq 1$ so the inductive hypothesis may be applied to $\mathfrak{B}$ and the theorem follows at once.

Formalization issues : a page in finite group theory

Fortunately a computer scientist is used to α-conversion:

Since $A/U = (K/U)(B/U)$, it follows that:

$$A = KB = DUB = DB$$

Also $K \cap B = U$ and $U\mho^p(K) \cap D = \mho^p(K)$, so

$$D \cap B = D \cap K \cap B = D \cap U = D \cap U\mho^p(K) \cap U = \mho^p(K) \cap U = 1$$

Thus $A = D \times B$. Then $K = D \times (B \times K) = D \times U$, so $D \simeq K/U$ is homocylic and $D/\Phi(D)$ is X-irreducible. Also $D \neq 1$ so the inductive hypothesis may be applied to B and the theorem follows at once.

Formalization issues : a page in finite group theory

Yet as usual, the devil is in the details:

Since $A/U = (K/U)(B/U)$, it follows that:

$$A = KB = DUB = DB$$

Also $K \cap B = U$ and $U\mho^p(K) \cap D = \mho^p(K)$, so

$$D \cap B = D \cap K \cap B = D \cap U = D \cap U\mho^p(K) \cap U = \mho^p(K) \cap U = 1$$

Thus $A = D \times B$. Then $K = D \times (B \times K) = D \times U$, so $D \simeq K/U$ is homocylic and $D/\Phi(D)$ is X-irreducible. Also $D \neq 1$ so the inductive hypothesis may be applied to B and the theorem follows at once.