Formalising the structure of extremal *p*-groups

Georges Gonthier

CENTRE DE RECHERCHE COMMUN
INRIA MICROSOFT RESEARCH

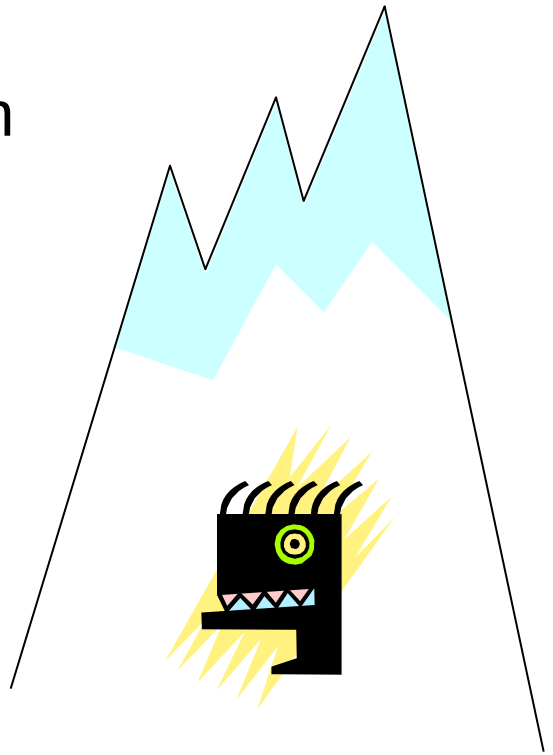
The Big Project

The Classification of Finite Simple Group

Frobenius groups
Thompson factorisation
character theory
linear representation
Galois theory
linear algebra
polynomials

Feit-Thompson

|G| odd
G simple
$G \approx F_p$



Goals and results

- Bring proof tools to mathematicians
- Apply software engineering to proof construction
- Prove landmark result
- Build on four-colour theorem experience

- Ssreflect proof language and Coq plugin
- Combinatorial components
- Linear algebra components
- Group theory components

4-color

Appel-Haken

finite groups

Feit-Thompson

Kepler

Hales

CENTRE DE RECHERCHE COMMUN

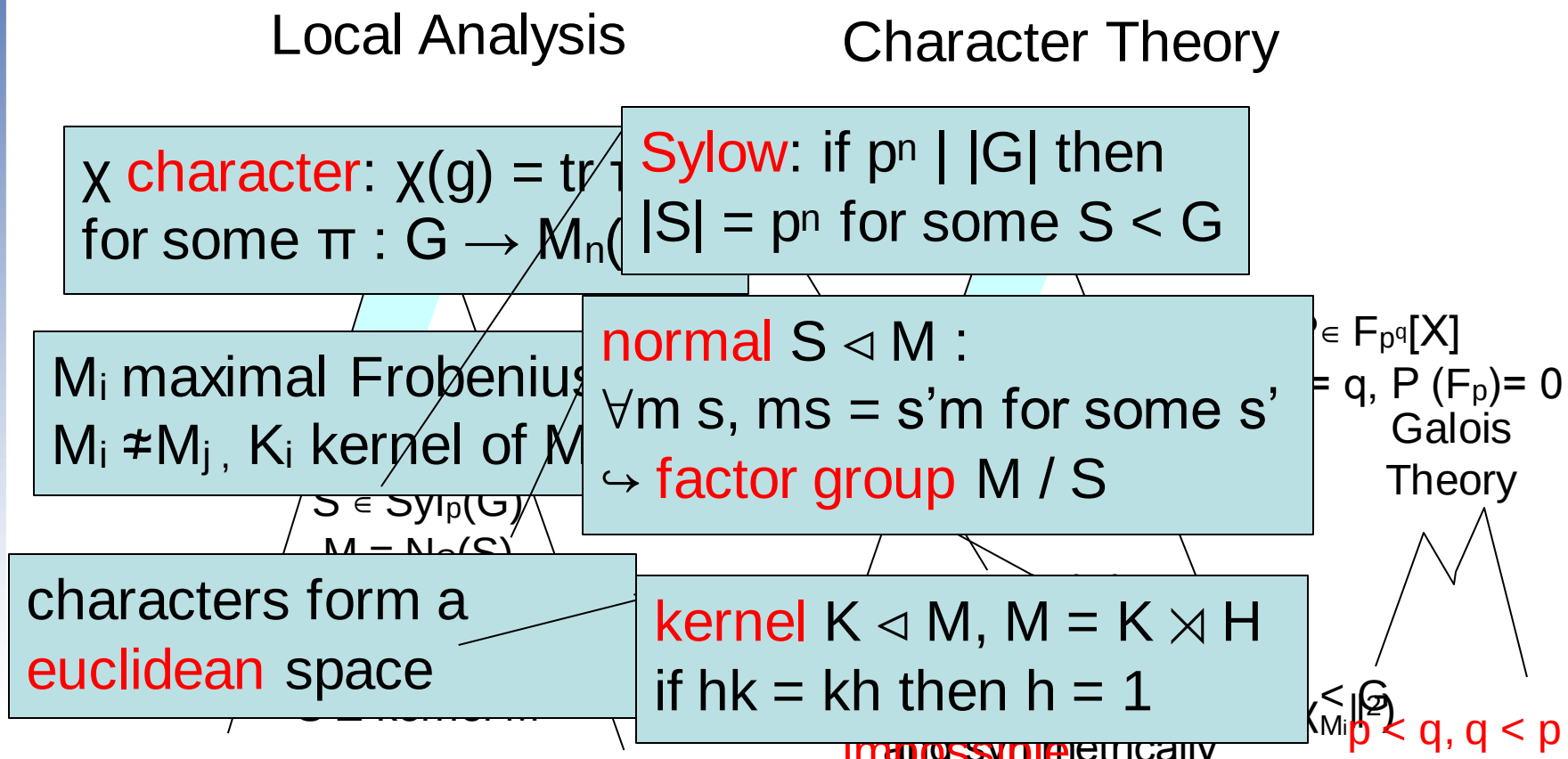
INRIA MICROSOFT RESEARCH

Tool review

- Data (inductive) types / propositions
- Computational reflection
  - compute values, types, and propositions
- Dependent types
  - first-class Structures
- Type / value inference
  - controlled by Coercion / Canonical Structure
- User notations

The Feit-Thompson proof

Let G be a minimal counter-example ...



Local Analysis

- Look at N(P), P a p-group, i.e. $P \rtimes N(P)$
- Use the nilpotent structure of P do prove stuff by induction on P and/or $A \subset N(P)$

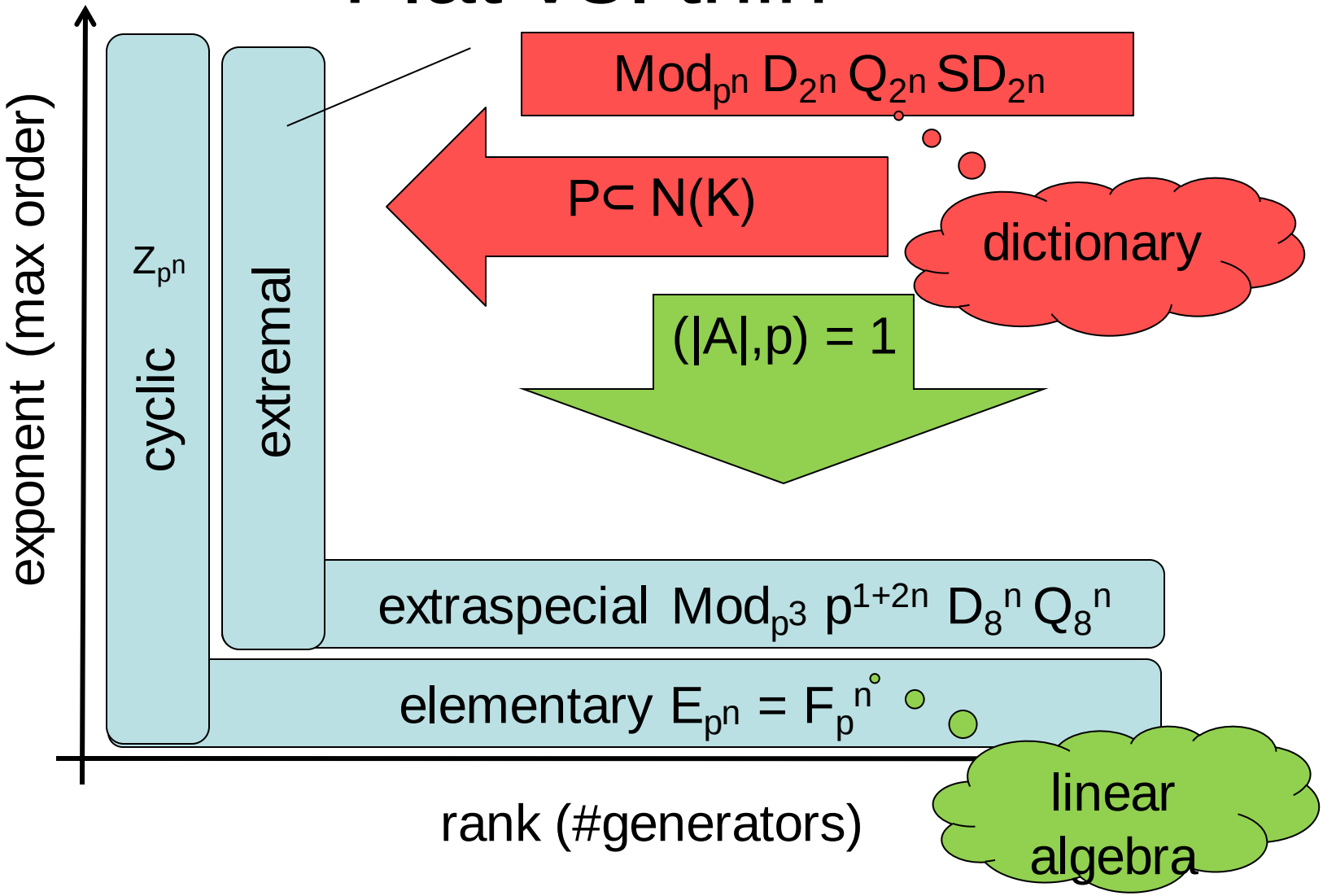
$$1 \lhd Z(P) \lhd Z_2(P) \lhd .... \lhd P$$

- Base cases: extraspecial and extremal p-groups.

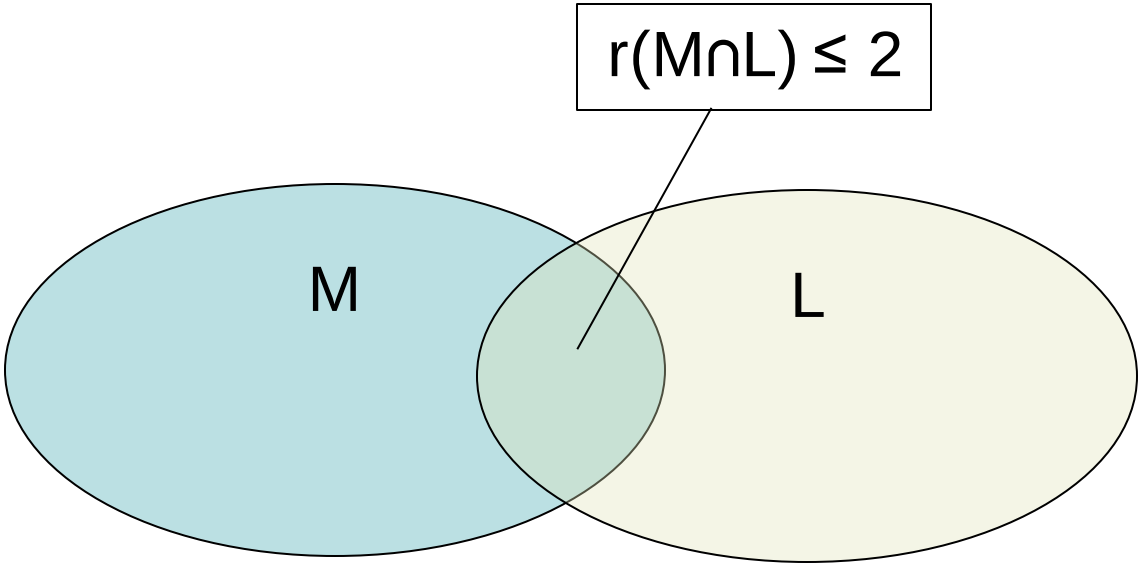
Roadmap

- Extremal p-groups
- Group presentations
- Big theorems

Flat vs. thin



The Uniqueness Theorem



Thompson transitivity:

$S \in Syl_p(G)$, $r(S) \geq 3$
maximal abelian $A \lhd S$
Q_1, Q_2 maximal in $\mathcal{U}(A, q)$
$Q_2 = Q_1^c$ for $c \in C_G(A)$

p-stability:

abelian $A \lhd S \in Syl_p(M)$
for some $C \lhd M$, $p \nmid |C|$
$A / C \lhd M / C$

Dictionary

```
Theorem dihedral2_structure :
  n > 1 -> extremal_generators G 2 n (x, y) -> G \isog 'D_m ->
[/\ [/\ X ><| Y = G, {in G :\: X, forall t, #[t] = 2}
      & {in X & G :\: X, forall z t, z ^ t = z^-1}],
    [/\ G ^`(1) = <[x ^+ 2]>, 'Phi(G) = G ^`(1), #|G^`(1)| = r
      & nil_class G = n.-1],
    'Ohm_1(G) = G /\ (forall k, k > 0 -> 'Mho^k(G) = <[x ^+ (2 ^ k)]>),
    [/\ yG :|: xyG = G :\: X, [disjoint yG & xyG]
      & forall M, maximal M G = pred3 X My Mxy M]
  & if n == 2 then (2.-abelem G : Prop) else
[/\ 'Z(G) = <[x ^+ r]>, #|'Z(G)| = 2,
     My \isog 'D_q, Mxy \isog 'D_q
   & forall U, cyclic U -> U \subset G -> #|G : U| = 2 -> U = X]].
```

Presentations

- $D_{2n} \cong$ Grp (x, y : $x^{2^{n-1}}$, y^2, x^y = x^-1)
- $SD_{8n} \cong$ Grp (x, y : x^{4n}, y^2, x^y = x^-1 * x^{2n-1})
- $Q_{2n} \cong$ Grp (x, y : x^{2n}, y^4, x^{2n-1} = y^2, x^y = x^-1)

... but this is not constructive (Post) !

```
Lemma Grp_dihedral :
 'D_m \isog Grp (x : y : (x ^+ q, y ^+ 2, x ^ y = x^-1)).
```

Faking free groups

- G ≈ Grp (x, y, ... : r, s, t)

  → (H ≲ G) ⇔ (H ≲ Grp (x, y, ... : r, s, t))
- H ≲ Grp (x, y, ... : r, s, t)

  → (H = ⟨x, y, ...⟩ & r = s = .. = 1)

```
Definition hom gT (B : {set gT}) p :=
  sat B 1 env1 (p (Cst 0)).
Definition iso gT (B : {set gT}) p :=
  forall rT (H : {group rT}), (H \homg B) = hom H p.
```

Faking relations

```
Inductive term :=
  | Cst of nat
  | Idx
  | Inv of term
  | Exp of term & nat
  | Mul of term & term
  | Conj of term & term
  | Comm of term & term.
```

```
Fixpoint eval {gT} e t : gT :=
 match t with
 | Cst i => nth 1 e i
 | Idx => 1
 | Inv t1 => (eval e t1)^-1
 | Exp t1 n => eval e t1 ^+ n
 | Mul t1 t2 => eval e t1 * eval e t2
 | Conj t1 t2 => eval e t1 ^ eval e t2
 | Comm t1 t2 => [~ eval e t1, eval e t2]
end.
```

```
Infix "*" := Mul : group_presentation.
Infix "^+" := Exp : group_presentation.
Infix "^" := Conj : group_presentation.
Notation "x ^-1" := (Inv x) : group_presentation.
```

Faking presentations

```
Inductive formula := Eq2 of term & term | And of formula & formula.
Inductive rel_type := NoRel | Rel vT of vT & vT.
Inductive type := Generator of term -> type | Formula of formula.
Inductive env gT := Env of {set gT} & seq gT.
```

```
Definition bool_of_rel r := if r is Rel vT v1 v2 then v1 == v2 else true.
Definition and_rel vT (v1 v2 : vT) r :=
  if r is Rel wT w1 w2 then Rel (v1, w1) (v2, w2) else Rel v1 v2.
 Fixpoint rel {gT} (e : seq gT) f r :=
  match f with
  | Eq2 s t => and_rel (eval e s) (eval e t) r
  | And f1 f2 => rel e f1 (rel e f2 r)
  end.
```

Using presentations

```
Lemma Grp'_dihedral :
  'D_m \isog Grp (x : y : (x ^+ 2, y ^+ 2, (x * y) ^+ q)).
Proof.
move=> gT G; rewrite Grp_dihedral.
apply/existsP/existsP=> [] [[x y]] /=.
   case/eqP=> <- xq1 y2 xy; exists (x * y, y).
   rewrite !xpair_eqE /= eqEsubset.
   ...
  by rewrite  y2 mulg1 xq1 !eqxx.
case/eqP=> <- x2 y2 xyq; exists (x * y, y).
....
by rewrite !mulgA mulgK -mulgA -!(expgS _ 1) x2 y2 mulg1.
Qed.
```

Proving presentations

```
Lemma isogEcard :
  forall rT aT (G : {group rT}) (H : {group aT}),
 (G \isog H) = (G \homg  H) && (#|H| <= #|G|).

Lemma card_dihedral : #|'D_m| = m.

Lemma normal_rank1_structure :
  forall gT p (G : {group gT}),
  p.-group G ->
  (forall X : {group gT}, X <| G -> abelian X -> cyclic X) ->
     cyclic G
 \/ [&& p == 2, extremal2 G & (#|G| >= 16) || (G \isog 'Q_8)].
```

Big Theorems

```
Theorem dihedral2_structure :
  n > 1 -> extremal_generators G 2 n (x, y) -> G \isog 'D_m ->
 [/\ [/\ X ><| y = G, {in G :\: X, forall t, #[t] = 2}
       & {in X & G :\: X, forall z t, z ^ t = z^-1}],
     [/\ G ^`(1) = <[x ^+ 2]>, 'Phi(G) = G ^`(1), #|G^`(1)| = r
       & nil_class G = n.-1],
     'Ohm_1(G) = G /\ (forall k, k > 0 -> 'Mho^k(G) = <[x ^+ (2 ^ k)]>),
     [/\ yG :|: xyG = G :\: X, [disjoint yG & xyG]
       & forall M, maximal M G = pred3 X My Mxy M]
  & if n == 2 then (2.-abelem G : Prop) else
 [/\ 'Z(G) = <[x ^+ r]>, #|'Z(G)| = 2,
      My \isog 'D_q, Mxy \isog 'D_q
   & forall U, cyclic U -> U \subset G -> #|G : U| = 2 -> U = X]].
```

Using big theorems

```
Lemma generators_2dihedral :
   n > 1 -> G \isog 'D_m ->
  exists2 xy, extremal_generators G 2 n xy
              & let: (x, y) := xy in #[y] = 2 /\ x ^ y = x^-1.


Lemma normal_rank1_structure :
   ....
have [[x y] genG _] := generators_2dihedral n_gt1 isoG.
have [_ _ _ [_ _ maxG]] := dihedral2_structure n_gt1 genG isoG.
```

Conclusions

- Formalizing big math sometimes requires big theorems.
- It is paramount to build usability into formal theories.
- Type theory and reflection are powerful and flexible tools engineer this usability.

CENTRE DE RECHERCHE COMMUN

INRIA
MICROSOFT RESEARCH