Gatterdam's proof that amalgamated free products are $\mathcal{E}^{n+1}$-computable

Christian Perfect

March 7, 2012

Theorem 0.1. *Let G_1, G_2 be $\mathcal{E}^n$-computable groups. Let H_1, H_2 be $\mathcal{E}^n$-decidable subgroups of the latter. Let $\phi' : H_1 \to H_2$ be an isomorphism, with ϕ', ϕ'^{-1} both $\mathcal{E}^n$-computable.*

Then the free product of G_1 and G_2 with H_1 and H_2 amalgamated, $G = G_1 \underset{\phi'}{*} G_2$, is $\mathcal{E}^{n+1}$-computable.

Proof. Let (i'_1, m'_1, j'_1) be the index of G_1 and (i'_2, m'_2, j'_2) be the index of G_2. The dashes will be explained later.

We can assume, without loss of generality, that $0 \notin i'_a(G_a)$ and $i'_a(1) = 1$ for $a = 1, 2$. CP (Might not even need this.)

By Magnus, Karrass, Solitar, etc., all elements $g \in G$ have normal form

$$g = hg_1 \dots g_r$$

where $h \in H_1$, and the g_i are coset representatives of G_a/H_a, $a = 1, 2$, such that $g_{i+1} \in G_1 \Leftrightarrow g_i \in G_2$.

The following proof becomes a lot easier if we redefine the factor group indices as follows:

$$\begin{aligned}
i_1(x) &:= 2i'_1(x), \\
m_1(x, y) &:= 2m'_1\left(\frac{x}{2}, \frac{y}{2}\right) \\
j_1(x) &:= 2j'_1\left(\frac{x}{2}\right).
\end{aligned}$$

$$
\begin{aligned}
i_2(x) &:= 2i_2'(x) - 1, \\
m_2(x,y) &:= 2m_2'\left(\frac{x+1}{2}, \frac{y+1}{2}\right) \\
j_2(x) &:= 2j_2'\left(\frac{x+1}{2}\right).
\end{aligned}
$$

Now,

$$
\begin{aligned}
x \in i_1(G_1) &\Leftrightarrow 2 \mid x \wedge \frac{x}{2} \in i_1'(G_1), \\
x \in i_1(H_1) &\Leftrightarrow x \in i_1(G_1) \wedge \frac{x}{2} \in i_1'(H_1), \\
x \in i_2(G_2) &\Leftrightarrow 2 \nmid x \wedge \frac{x+1}{2} \in i_2'(G_2), \\
x \in i_2(H_2) &\Leftrightarrow x \in i_2(G_2) \wedge \frac{x+1}{2} \in i_2'(H_2).
\end{aligned}
$$

The subgroup isomorphism also needs to be redefined:

$$
\begin{aligned}
\phi(x) &:= 2\phi'\left(\frac{x}{2}\right) - 1, \\
\phi^{-1}(x) &:= 2\phi'^{-1}\left(\frac{x+1}{2}\right).
\end{aligned}
$$

In order to do multiplication, we need to be able to split every $g_a \in G_a$, $a = 1, 2$, into a word of the form $h_a k_a$, where $h_a \in H_a$ and k_a is a coset representative of g_a in G_a/H_a.

Define:

$$
\begin{aligned}
k_a(x) &:= \min_{y \leq x} \left(m_a(x, j_a(y)) \in i_a(H_a)\right), \\
h_a(x) &:= m_a(x, j_a(k_a(x))).
\end{aligned}
$$

Now we can define $i(G)$ by

$$
i(hg_1 \dots g_r) := [h, g_1, \dots, g_r].
$$

(encode it as a Gödel number)

$i(G)$ is $\mathcal{E}^n$-decidable because $x \in i(G)$ if and only if:

- x is a Gödel number
- $(x)_0 \in i_1(H_1)$
- $\forall 1 < i < |x|$, $(x)_i \in i_1(G_1) \Leftrightarrow (x)_{i-1} \in i_2(G_2)$ CP (check this is true –

different from Gatterdam's) and $(x)_i \in i_a(G_a) \to k_a((x)_i) = (x)_0$. CP Gatterdam uses h_a but I think it should be k_a

Now define a function $r : (i_1(G_1) \cup i_2(G_2)) \times i(G) \to i(G)$ which does multiplication of elements of G on the left by elements of G_1 and G_2.

r is defined by cases in a decision tree, so here's some programming-ish syntax:

$r(x, y) :=$

if $x \in i_1(G_1)$ **then**
  if $x \in i_1(H_1)$ **then**
    $[m_1(x, (y)_0)] +\!\!+ y[1 \ldots]$
  else
    if $(y)_1 \in i_1(G_1)$ **then**
      if $m_1(m_1(x, (y)_0), (y)_1) \in i_1(H_1)$ **then**
        $[m_1(m_1(x, (y)_0), (y)_1)] +\!\!+ y[2 \ldots]$
      else
        $[h_1 m_1(m_1(x, (y)_0), (y)_1), k_1 m_1(m_1(x, (y)_0), (y)_1)] +\!\!+ y[2 \ldots]$
      end if
    else
      $[h_1 m_1(x, (y)_0), k_1 m_2(x, (y)_0)] +\!\!+ y[1 \ldots]$
    end if
  end if
else
  if $x \in i_2(H_2)$ **then**
    $[m_1(\phi^{-1}(x), (y)_0)] +\!\!+ y[1 \ldots]$
  else
    if $(y)_1 \in i_2(G_2)$ **then**
      if $m_2(m_2(x, \phi((y)_0)), (y)_1) \in i_2(H_2)$ **then**
        $[\phi^{-1} m_2(m_2(x, \phi((y)_0)), (y)_1)] +\!\!+ y[2 \ldots]$
      else
        $[\phi^{-1} h_2 m_2(m_2(x, \phi((y)_0)), (y)_1), k_2 m_2(m_2(x, \phi((y)_0)), (y)_1)] +\!\!+ y[2 \ldots]$
      end if
    else
      $[\phi^{-1} h_2 m_2(x, \phi((y)_0)), k_2 m_2(x, \phi((y)_0))] +\!\!+ y[1 \ldots]$
    end if
  end if
end if

Hopefully it's clear that this can be rewritten unambiguously as a definition-by-cases in the usual format.

Now we can do multiplication on G in general:

$$m([], y) := y,$$
$$m(x : xs, y) := r(x, m(xs, y)).$$

And inversion:

$$j([1]) := [1],$$
$$j(x : xs) := r(\hat{j}(x), j(xs)).$$

$$\hat{j}(x) := \begin{cases} j_1(x), & x \in i_1(G_1), \\ j_2(x), & x \in i_2(G_2). \end{cases}$$

$\square$

($x : xs$ means a list of the form $[x, x_1, x_2, \ldots]$. I borrowed this syntax from Haskell. I need to show that you can define recursive functions on lists like this, but I hope you agree it works)

Gatterdam then also shows that the embeddings $G_a \to G$ are natural, but I lost the will.