```
POLYNOM_3
---------
*C polynom_3_begin            ************ POLYNOM_3 ************
*C omral_com   ============================
               ORDERED-MONOID RING A-LISTS
               ============================
               omral = o(rdered) m(onoid) a-l(ist)
*C omral_com_1 ========================
               DEFINITION OF OMRAL TYPE
               ========================
*D omralist_df                 omral(<g:g:*>;<r:r:*>)== omralist{}(<g>; <r>)
*A omralist                    omral(g;r) ==  oal(g↓oset;r↓+gp)
*T omralist_wf            0 2 ∀g:OCMon. ∀r:CRng.  omral(g;r) ∈ DSet
*M omralist_ml                 note_reduction_strength ‘omralist‘ ‘8‘;;
*D omral_dom_df
               dom{<g:g:*>,<r:r:*>}(<ps:ps:*>)== omral_dom{}(<g>; <r>; <ps>)
               dom(<ps:ps:*>)== omral_dom{}(<g>; <r>; <ps>)
*A omral_dom                   dom(ps) ==  dom(ps)
*T omral_dom_wf           0 2 ∀g:OCMon. ∀r:CRng. ∀ps:(|g| × |r|) List.  dom(ps) ∈ MSet{g↓oset}
*T omral_dom_wf2          0 3 ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.  dom(ps) ∈ FSet{g↓oset}
*M omral_dom_eval
               let omral_dom_nilC =
                 MacroC ‘omral_dom_nilC‘
                 (EvalC ‘‘omral_dom‘‘)
                 ⌈dom([])⌉
                 IdC
                 ⌈0{g↓oset}⌉
               ;;
               let omral_dom_cons_prC =
                 MacroC ‘omral_cons_prC‘
                 (EvalC ‘‘omral_dom‘‘)
                 ⌈dom(<k, v>::ps)⌉
                 (UnfoldC ‘omral_dom‘)
                 ⌈mset_inj{g↓oset}(k) + dom(ps)⌉
               ;;
               add_AbReduce_conv ‘omral_dom‘
                 (omral_dom_nilC ORELSEC omral_dom_cons_prC)
               ;;
*T omralist_car_properties 2 2
               ∀g:OCMon. ∀r:CRng. ∀ws:|omral(g;r)|.
                 ↑sd_ordered(map(λx.x.1;ws)) ∧ ¬↑(0 ∈b map(λx.x.2;ws))
*M oal_to_omral
               % Lifting Theorems from oalists to omralists %
               let omral_opids =
               ‘‘omralist omral_plus omral_dom grp_lt grp_leq grp_blt
                 omral_zero omral_minus omral_inj‘‘
               ;;
               let OmRalC =
                ForceReduceC ‘5‘ ANDTHENC TryC (FoldsC omral_opids) ;;
               let OmRalCStr =
                 "ForceReduceC ‘5‘ ANDTHENC TryC (FoldsC ‘‘"
                 J
                 concatenate_strings
                   (map (\id.tok_to_string id J " ") omral_opids)
                 J
                 "‘‘)"
               ;;
```

```
              let mk_omral_thm old_name new_name new_pos =
               add_specialized_theorem
                old_name
                [‘g’,⌈OCMon⌉;‘r’,⌈CRng⌉]     % New outer context %
                [⌈parm{i}⌉;⌈g↓oset⌉;⌈r↓+gp⌉] % Bindings for outer context of old thm %
                OmRalC
                OmRalCStr
                new_name
                new_pos
                ; refresh()
              ;;
*T rng_before_imp_before_all 2 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀ps:|omral(g;r)|.
                ↑before(k;map(λz.z.1;ps)) ⇒ ↑(∀b x(:|g|) ∈ map(λz.z.1;ps). x <b k)
*T rng_before_all_imp_before 2 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀ps:(|g| × |r|) List.
                ↑(∀b x(:|g|) ∈ map(λz.z.1;ps). x <b k) ⇒ ↑before(k;map(λz.z.1;ps))
*T omralist_cases         2 2
              ∀g:OCMon. ∀r:CRng. ∀Q:|omral(g;r)| → ℙ.
                Q[[]]
                ⇒ (∀ws:|omral(g;r)|. ∀x:|g|. ∀y:|r|.
                      ↑before(x;map(λx.x.1;ws)) ⇒ ¬(y = 0) ⇒ Q[<x, y>::ws])
                ⇒ {∀ws:|omral(g;r)|. Q[ws]}
*T omralist_ind_a         2 2
              ∀g:OCMon. ∀r:CRng. ∀Q:|omral(g;r)| → ℙ.
                Q[[]]
                ⇒ (∀ws:|omral(g;r)|
                      Q[ws]
                      ⇒ (∀x:|g|. ∀y:|r|.
                            ↑before(x;map(λx.x.1;ws)) ⇒ ¬(y = 0) ⇒ Q[<x, y>::ws]))
                ⇒ {∀ws:|omral(g;r)|. Q[ws]}
*T omral_lookups_same_a   2 2
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|.  (∀u:|g|. ps[u] = qs[u]) ⇒ ps = qs
*T rng_lookup_before_start 2 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀ps:|omral(g;r)|.
                ↑before(k;map(λz.z.1;ps)) ⇒ ps[k] = 0
*T lookup_omral_eq_zero   2 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀ps:|omral(g;r)|.  ¬↑(k ∈b dom(ps)) ⇒ ps[k] = 0
*C omral_plus_com
              ===================
              OMRAL PLUS FUNCTION
              ===================
              Lifting of oal merge function
*D omral_plus_df
              Parens ::Prec(inop)::
                <ps:ps:L> ++<g:g:L>,<r:r:L> <qs:qs:L>
                == omral_plus{}(<g>; <r>; <ps>; <qs>)
              Parens ::Prec(inop)::
                <ps:ps:L> ++ <qs:qs:L>
                == omral_plus{}(<g>; <r>; <ps>; <qs>)
*A omral_plus                ps ++ qs ==  ps ++ qs
*T omral_plus_wf          1 3
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:(|g| × |r|) List.  ps ++ qs ∈ (|g| × |r|) List
*T omral_plus_sd_ordered  2 2
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:(|g| × |r|) List.
                ↑sd_ordered(map(λx.x.1;ps))
                ⇒ ↑sd_ordered(map(λx.x.1;qs))
```

```
                ⇒ ↑sd_ordered(map(λx.x.1;ps ++ qs))
*T omral_plus_non_zero_vals 2 2
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:(|g| × |r|) List.
                ¬↑(0 ∈b map(λx.x.2;ps))
                ⇒ ¬↑(0 ∈b map(λx.x.2;qs))
                ⇒ ¬↑(0 ∈b map(λx.x.2;ps ++ qs))
*T omral_plus_wf2         2 2 ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|.  ps ++ qs ∈ |omral(g;r)|
*T omral_plus_dom         2 2
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|.  ↑(dom(ps ++ qs) ⊆b dom(ps) ∪ dom(qs))
*T lookup_omral_plus      2 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀ps,qs:|omral(g;r)|.  (ps ++ qs)[k] = ps[k] +r qs[k]
*T omral_plus_comm        1 2 ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|.  ps ++ qs = qs ++ ps
*T omral_plus_assoc       1 2
              ∀g:OCMon. ∀r:CRng. ∀ps,qs,rs:|omral(g;r)|.  ps ++ (qs ++ rs) = (ps ++ qs) ++ rs
*C omral_zmi_com
              ========================================
              OMRAL ZERO, MINUS AND INJECTION FUNCTIONS
              ========================================
              All lifted from oal development.
*D omral_zero_df              00<g:g:*>,<r:r:*>== omral_zero{}(<g>; <r>)
*A omral_zero                 00g,r ==  00
*T omral_zero_wf          0 2 ∀g:OCMon. ∀r:CRng.  00g,r ∈ |omral(g;r)|
*D omral_minus_df
              Parens ::Prec(preop)::
                --<g:g:L>,<r:r:L> <ps:ps:L>
                == omral_minus{}(<g>; <r>; <ps>)
              Parens ::Prec(preop)::  --<ps:ps:L>== omral_minus{}(<g>; <r>; <ps>)
*A omral_minus                --ps ==  --ps
*T omral_minus_wf         0 4 ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.  --ps ∈ |omral(g;r)|
*D omral_inj_df
              inj{<g:g:*>,<r:r:*>}(<k:k:*>,<v:v:*>)== omral_inj{}(<g>; <r>; <k>; <v>)
              inj(<k:k:*>,<v:v:*>)== omral_inj{}(<g>; <r>; <k>; <v>)
*A omral_inj                  inj(k,v) ==  inj(k,v)
*T omral_inj_wf           0 3 ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|.  inj(k,v) ∈ |omral(g;r)|
*T omral_dom_inj          1 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|.
                dom(inj(k,v)) = if v =b 0 then 0{g↓oset} else mset_inj{g↓oset}(k) fi
*T lookup_omral_inj       1 2
              ∀g:OCMon. ∀r:CRng. ∀k,k':|g|. ∀v:|r|.  inj(k,v)[k'] = when k =b k'. v
*T comb_for_omral_inj_wf  0 0
              (λg,r,k,v,z.inj(k,v)) ∈ g:OCMon
                                    → r:CRng
                                    → k:|g|
                                    → v:|r|
                                    → ↓True
                                    → |omral(g;r)|
*T omral_fact             1 2
              ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.
                ps = msFor{oal_mon(g↓oset;r↓+gp)} k' ∈ dom(ps). inj(k',ps[k'])
*T omral_fact_a           2 2
              ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.
                ps = msFor{omral_alg(g;r)↓grp} k' ∈ dom(ps). inj(k',ps[k'])
*C omral_scale_com
              ======================
              OMRAL SCALING FUNCTION
              ======================
              Scales keys and values of an omralist.
```

```
*D omral_scale_df
              Parens ::Prec(preop)::
                <<k:k:*>,<v:v:*>>*<g:mon:L>,<r:rng:L> <ps:ps:E>
                == omral_scale{}(<g>; <r>; <k>; <v>; <ps>)
              Parens ::Prec(preop)::
                <<k:k:*>,<v:v:*>>* <ps:ps:E>
                == omral_scale{}(<g>; <r>; <k>; <v>; <ps>)
*M omral_scale_ml
              <k,v>* ps
              ==r case ps of
                    [] => []
                    p::ps' => if (v * p.2) =b 0
                              then <k,v>* ps'
                              else <k * p.1, v * p.2>::(<k,v>* ps')
                              fi
                  esac
*M omral_scale_eval
              let omral_scale_nilC =
                FwdMacroC ‘omral_scale_nilC‘
                (RecEvalC ‘‘omral_scale‘‘) ⌜<k,v>* []⌝ ;;
              let omral_scale_cons_prC =
                FwdMacroC ‘omral_scale_cons_prC‘
                (RecEvalC ‘‘omral_scale‘‘) ⌜<k,v>* (<k', v'>::ps)⌝ ;;
              add_AbReduce_conv ‘omral_scale‘
                (omral_scale_nilC ORELSEC omral_scale_cons_prC) ;;
*T omral_scale_wf         3 2
              ∀g:GrpSig. ∀r:RngSig. ∀k:|g|. ∀v:|r|. ∀ps:(|g| × |r|) List.
                <k,v>* ps ∈ (|g| × |r|) List
*T omral_scale_dom_pred   3 4
              ∀g:OCMon. ∀r:CRng. ∀Q:|g| → 𝔹. ∀k:|g|. ∀v:|r|. ∀ps:(|g| × |r|) List.
                ↑(∀b x(:|g|) ∈ map(λz.z.1;ps). Q[k * x])
                ⇒ ↑(∀b x(:|g|) ∈ map(λz.z.1;<k,v>* ps). Q[x])
*T omral_dom_scale        4 6
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|. ∀ps:|omral(g;r)|.
                ↑(dom(<k,v>* ps) ⊆b fs-map(λk'.k' * k, dom(ps)))
*T omral_scale_dom_bound  3 5
              ∀g:OCMon. ∀r:CRng. ∀bound,k:|g|. ∀v:|r|. ∀ps:(|g| × |r|) List.
                ↑(∀b x(:|g|) ∈ map(λz.z.1;ps). x <b bound)
                ⇒ ↑(∀b x(:|g|) ∈ map(λz.z.1;<k,v>* ps). x <b k * bound)
*C omral_scale_sd_ordered_com
              The proof here needs some cleaning up.
              Probably, worth pulling out the second
              induction and generalizing it a bit.
*T omral_scale_sd_ordered 5 6
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|. ∀ps:(|g| × |r|) List.
                ↑sd_ordered(map(λz.z.1;ps)) ⇒ ↑sd_ordered(map(λz.z.1;<k,v>* ps))
*T omral_scale_non_zero_vals 3 5
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|. ∀ps:(|g| × |r|) List.
                ¬↑(0 ∈b map(λx.x.2;ps)) ⇒ ¬↑(0 ∈b map(λx.x.2;<k,v>* ps))
*T omral_scale_wf2        2 4
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|. ∀ps:|omral(g;r)|.  <k,v>* ps ∈ |omral(g;r)|
*T lookup_omral_scale_a   4 6
              ∀g:OCMon. ∀r:CRng. ∀k,k':|g|. ∀v:|r|. ∀ps:|omral(g;r)|.
                (<k,v>* ps)[k * k'] = v * ps[k']
*T lookup_omral_scale_b   3 6
              ∀g:OCMon. ∀r:CRng. ∀k,k':|g|. ∀v:|r|. ∀ps:(|g| × |r|) List.
                ¬(∃d:|g|. ↑(d ∈b dom(ps)) ∧ k * d = k') ⇒ (<k,v>* ps)[k'] = 0
```

```
*T lookup_omral_scale_c   4 6
              ∀g:OCMon. ∀r:CRng. ∀z,k:|g|. ∀v:|r|. ∀ps:|omral(g;r)|.
                (<k,v>* ps)[z] = msFor{r↓+gp} y ∈ dom(ps). when (k * y) =b z. v * ps[y]
*T lookup_omral_scale_d   1 0
              ∀g:OCMon. ∀r:CRng. ∀z,k:|g|. ∀v:|r|. ∀ps:|omral(g;r)|.
                (<k,v>* ps)[z] = (Σy ∈ dom(ps). when (k * y) =b z. v * ps[y])
*C omral_times_com
              ====================
              OMRAL TIMES FUNCTION
              ====================
*D omral_times_df
              Parens ::Prec(inop)::
                <ps:ps:L> **<g:g:L>,<r:r:L> <qs:qs:L>
                == omral_times{}(<g>; <r>; <ps>; <qs>)
              Parens ::Prec(inop)::
                <ps:ps:L> ** <qs:qs:L>
                == omral_times{}(<g>; <r>; <ps>; <qs>)
*M omral_times_ml
              ps ** qs==r case ps of [] => [] | p::ps' => <p.1,p.2>* qs ++ (ps' ** qs) esac
*M omral_times_eval
              let omral_times_nilC =
                FwdMacroC `omral_times_nilC`
                (RecEvalC ``omral_times``) ⌜[] ** qs⌝ ;;
              let omral_times_cons_prC =
                FwdMacroC `omral_times_cons_prC`
                (RecEvalC ``omral_times``) ⌜(<k, v>::ps) ** qs⌝
              ;;
              add_AbReduce_conv `omral_times`
                (omral_times_nilC ORELSEC omral_times_cons_prC)
              ;;
*T omral_times_wf         2 2
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:(|g| × |r|) List.  ps ** qs ∈ (|g| × |r|) List
*T omral_times_sd_ordered 2 3
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:(|g| × |r|) List.
                ↑sd_ordered(map(λz.z.1;qs)) ⇒ ↑sd_ordered(map(λz.z.1;ps ** qs))
*T omral_times_non_zero_vals 2 3
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:(|g| × |r|) List.
                ¬↑(0 ∈b map(λx.x.2;qs)) ⇒ ¬↑(0 ∈b map(λx.x.2;ps ** qs))
*T omral_times_wf2        2 4 ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|.  ps ** qs ∈ |omral(g;r)|
*T lookup_omral_times     4 6
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|. ∀z:|g|.
                (ps ** qs)[z]
                = msFor{r↓+gp} x ∈ dom(ps)
                    msFor{r↓+gp} y ∈ dom(qs). when (x * y) =b z. ps[x] * qs[y]
*T lookup_omral_times_a   0 0
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|. ∀z:|g|.
                (ps ** qs)[z] = (Σx ∈ dom(ps). Σy ∈ dom(qs). when (x * y) =b z. ps[x] * qs[y])
*T mset_on_grp_eq         2 0 ∀g:OCMon. MSet{g↓set} = MSet{g↓oset}
*T mset_inc               2 0 ∀g:OCMon. MSet{g↓set} ⊆ MSet{g↓oset}
*T mset_inc_a             2 1 ∀g:OCMon. MSet{g↓oset} ⊆ MSet{g↓set}
*T omral_times_dom        5 8
              ∀g:OCMon. ∀r:CRng. ∀ps,qs:|omral(g;r)|.  ↑(dom(ps ** qs) ⊆b dom(ps) × dom(qs))
*T omral_times_assoc      5 9 ∀g:OCMon. ∀a:CRng.  Assoc(|omral(g;a)|;λps,qs.ps ** qs)
*T omral_times_assoc_a    1 0
              ∀g:OCMon. ∀a:CRng. ∀ps,qs,rs:|omral(g;a)|.  ps ** (qs ** rs) = (ps ** qs) ** rs
#T omral_times_assoc_b    4 8
              ∀g:OCMon. ∀a:CRng. ∀ps,qs,rs:|omral(g;a)|.  ps ** (qs ** rs) = (ps ** qs) ** rs
```

```
*T omral_times_comm       3 5 ∀g:OCMon. ∀a:CRng.  Comm(|omral(g;a)|;λps,qs.ps ** qs)
*T omral_times_comm_a     1 0 ∀g:OCMon. ∀a:CRng. ∀ps,qs:|omral(g;a)|.  ps ** qs = qs ** ps
*T omral_bilinear         5 7
              ∀g:OCMon. ∀a:CRng.  BiLinear(|omral(g;a)|;λps,qs.ps ++ qs;λps,qs.ps ** qs)
*T omral_bilinear_a       1 0
              ∀g:OCMon. ∀a:CRng. ∀ps,qs,rs:|omral(g;a)|.
                ps ** (qs ++ rs) = (ps ** qs) ++ (ps ** rs)
                ∧ (qs ++ rs) ** ps = (qs ** ps) ++ (rs ** ps)
*C omral_one_act_com
              ====================
              OMRAL ONE AND ACTION
              ====================
*D omral_one_df
              Parens ::Prec(preop)::  11<g:g:L>,<r:r:L>== omral_one{}(<g>; <r>)
              11== omral_one{}(<g>; <r>)
*A omral_one                  11 ==  inj(e,1)
*T omral_one_wf           0 0 ∀g:OCMon. ∀r:CRng.  11 ∈ |omral(g;r)|
*T omral_dom_one          2 3 ∀g:OCMon. ∀r:CRng.  ¬(0 = 1) ⇒ dom(11) = mset_inj{g↓oset}(e)
*D omral_action_df
              Parens ::Prec(inop)::
                <v:v:L> ··<g:g:L>,<r:r:L> <ps:ps:L>
                == omral_action{}(<g>; <r>; <v>; <ps>)
              Parens ::Prec(inop)::
                <v:v:L> ·· <ps:ps:L>
                == omral_action{}(<g>; <r>; <v>; <ps>)
*A omral_action               v ·· ps ==  <e,v>* ps
*T omral_action_wf        0 0
              ∀g:OCMon. ∀r:CRng. ∀v:|r|. ∀ps:|omral(g;r)|.  v ·· ps ∈ |omral(g;r)|
*T comb_for_omral_action_wf 0 0
              (λg,r,v,ps,z.v ·· ps) ∈ g:OCMon
                                    → r:CRng
                                    → v:|r|
                                    → ps:|omral(g;r)|
                                    → ↓True
                                    → |omral(g;r)|
*C omral_dom_action_com
              Nice simple example of monotonicity
              reasoning here.
*T omral_dom_action       3 4
              ∀g:OCMon. ∀r:CRng. ∀v:|r|. ∀ps:|omral(g;r)|.  ↑(dom(v ·· ps) ⊆b dom(ps))
*T lookup_omral_action    2 3
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v:|r|. ∀ps:|omral(g;r)|.  (v ·· ps)[k] = v * ps[k]
*C omral_alg_com
              ====================================
              ASSEMBLY OF OMRAL FREE MONOID ALGEBRA
              ====================================
*T omral_times_ident_r    2 1 ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.  ps ** 11 = ps
*T omral_times_ident_l    4 6 ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.  11 ** ps = ps
*T omral_action_one       1 3 ∀g:OCMon. ∀r:CRng. ∀ps:|omral(g;r)|.  1 ·· ps = ps
*T omral_action_times     1 3
              ∀g:OCMon. ∀r:CRng. ∀v,w:|r|. ∀ps:|omral(g;r)|.  (v * w) ·· ps = v ·· (w ·· ps)
*T omral_action_times_r1  4 7
              ∀g:OCMon. ∀r:CRng. ∀v:|r|. ∀ps,qs:|omral(g;r)|.
                v ·· (ps ** qs) = (v ·· ps) ** qs
*T omral_action_times_r2  1 2
              ∀g:OCMon. ∀r:CRng. ∀v:|r|. ∀ps,qs:|omral(g;r)|.
                v ·· (ps ** qs) = ps ** (v ·· qs)
```

```
*T omral_action_plus_l    1 3
              ∀g:OCMon. ∀r:CRng. ∀v,w:|r|. ∀ps:|omral(g;r)|.
                (v +r w) ·· ps = (v ·· ps) ++ (w ·· ps)
*T omral_action_plus_r    1 4
              ∀g:OCMon. ∀r:CRng. ∀v:|r|. ∀ps,qs:|omral(g;r)|.
                v ·· (ps ++ qs) = (v ·· ps) ++ (v ·· qs)
*T omral_action_inj       2 2
              ∀g:OCMon. ∀r:CRng. ∀k:|g|. ∀v,v':|r|.  v ·· inj(k,v') = inj(k,v * v')
*D omral_alg_df               omral_alg(<g:g:*>;<r:r:*>)== omral_alg{}(<g>; <r>)
*A omral_alg   omral_alg(g;r) ==
                <|omral(g;r)|
                , =b
                , λx,y.tt
                , λx,y.x ++ y
                , 00g,r
                , λx.--x
                , λx,y.x ** y
                , 11
                , λx,y.(inr · )
                , λa,x.a ·· x>
*T omral_alg_wf           0 4 ∀g:OCMon. ∀r:CRng.  omral_alg(g;r) ∈ AlgebraSig(|r|)
*T omral_alg_wf2          5 5 ∀g:OCMon. ∀r:CRng.  omral_alg(g;r) ∈ r-CAlgebra
*T omral_inj_mon_op       3 6
              ∀g:OCMon. ∀r:CRng. ∀k,k':|g|.  inj(k * k',1) = inj(k,1) ** inj(k',1)
*D omral_alg_umap_df
              alg_umap{<g:mon:*>,<a:rng:*>}(<n:alg:*>,<f:mon->alg:*>)
              == omral_alg_umap{}(<g>; <a>; <n>; <f>)
              alg_umap(<n:alg:*>,<f:mon->alg:*>)== omral_alg_umap{}(<g>; <a>; <n>; <f>)
*A omral_alg_umap             alg_umap(n,f) ==  λps:|omral(g;a)|. Σk ∈ dom(ps). ps[k] ·n (f k)
*T omral_alg_umap_wf      0 3
              ∀g:OCMon. ∀a:CRng. ∀n:a-Algebra. ∀f:|g| ⟶ |n|.
                alg_umap(n,f) ∈ |omral(g;a)| ⟶ |n|
*T omral_alg_umap_is_hom  6 8
              ∀g:OCMon. ∀a:CRng. ∀n:a-Algebra. ∀f:MonHom(g,n↓rg↓xmn).
                IsAlgHom{a,omral_alg(g;a),n}(alg_umap(n,f))
*T omral_alg_umap_tri_comm 3 5
              ∀g:OCMon. ∀a:CRng. ∀n:a-Algebra. ∀f:|g| ⟶ |n|.
                alg_umap(n,f) o (λk.inj(k,1)) = f
*T omral_alg_umap_unique  4 7
              ∀g:OCMon. ∀a:CRng. ∀n:a-Algebra. ∀f:|g| ⟶ |n|.
              ∀f':a-AlgebraHom(omral_alg(g;a);n).
                f' o (λk:|g|. inj(k,1)) = f ⇒ f' = alg_umap(n,f)
*D omral_fma_df               omral_fma(<g:g:*>;<a:a:*>)== omral_fma{}(<g>; <a>)
*A omral_fma   omral_fma(g;a) ==  <omral_alg(g;a), λk.inj(k,1), λn,f.alg_umap(n,f)>
*T omral_fma_wf           0 4 ∀g:OCMon. ∀a:CRng.  omral_fma(g;a) ∈ FMASig(g;a)
*T omral_fma_wf2          4 5 ∀g:OCMon. ∀a:CRng.  omral_fma(g;a) ∈ FMonAlg(g;a)
*C polynom_3_end              ************************************
Thm stats: <log2 (# pscript lines)> <log2 (expansion time in sec)>
```