The Localization of a Commutative Ring

Anthony Bordg

February 6, 2026

Abstract

We formalize the localization [1, II, §4] of a commutative ring R with respect to a multiplicative subset (i.e. a submonoid of R seen as a multiplicative monoid).
This localization is itself a commutative ring and we build the natural homomorphism of rings from R to its localization.

Contents

theory *Localization*
imports *Main HOL−Algebra.Group HOL−Algebra.Ring HOL−Algebra.AbelCoset*
begin

Contents:

- We define the localization of a commutative ring R with respect to a multiplicative subset, i.e. with respect to a submonoid of R (seen as a multiplicative monoid), cf. [*rec-rng-of-frac*].
- We prove that this localization is a commutative ring (cf. [*crng-rng-of-frac*]) equipped with a homomorphism of rings from R (cf. [*rng-to-rng-of-frac-is-ring-hom*]).

1 The Localization of a Commutative Ring

1.1 Localization

locale *submonoid* = *monoid M* **for** *M* (**structure**) +
fixes *S*
assumes *subset* : $S \subseteq$ *carrier M*

and *m-closed* [*intro*, *simp*] : $[\![x \in S; y \in S]\!] \Longrightarrow x \otimes y \in S$
and *one-closed* [*simp*] : $\mathbf{1} \in S$

lemma (**in** *submonoid*) *is-submonoid*: *submonoid M S*
by (*rule submonoid-axioms*)

locale *mult-submonoid-of-rng* = *ring R* + *submonoid R S* **for** *R* **and** *S*

locale *mult-submonoid-of-crng* = *cring R* + *mult-submonoid-of-rng R S* **for** *R* **and** *S*

locale *eq-obj-rng-of-frac* = *cring R* + *mult-submonoid-of-crng R S* **for** *R* (**structure**) **and** *S* +
fixes *rel*
defines $rel \equiv (\!|carrier = carrier\ R \times S,\ eq = \lambda(r,s)\ (r',s').\ \exists t{\in}S.\ t \otimes ((s' \otimes r) \ominus (s \otimes r')) = \mathbf{0}|\!)$

lemma (**in** *abelian-group*) *minus-to-eq* :
assumes *abelian-group G* **and** $x \in carrier\ G$ **and** $y \in carrier\ G$ **and** $x \ominus y = \mathbf{0}$
shows $x = y$
by (*metis add.inv-solve-right assms(2) assms(3) assms(4) l-zero minus-eq zero-closed*)

lemma (**in** *eq-obj-rng-of-frac*) *equiv-obj-rng-of-frac*:
shows *equivalence rel*
proof
show $\bigwedge x.\ x \in carrier\ rel \Longrightarrow x\ .{=}_{rel}\ x$
proof−
fix *x*
assume $x \in carrier\ rel$
then have $f1{:}\mathbf{1} \otimes ((snd\ x \otimes fst\ x) \ominus (snd\ x \otimes fst\ x)) = \mathbf{0}$
using *rel-def subset l-one minus-eq add.r-inv rev-subsetD*
by *auto*
moreover have $x = (fst\ x,\ snd\ x)$
by *simp*
thus $x\ .{=}_{rel}\ x$
using *rel-def one-closed f1*
by *auto*
qed
show $\bigwedge x\ y.\ x\ .{=}_{rel}\ y \Longrightarrow x \in carrier\ rel \Longrightarrow y \in carrier\ rel \Longrightarrow y\ .{=}_{rel}\ x$
proof−
fix *x y*
assume $a1{:}x\ .{=}_{rel}\ y$ **and** $a2{:}x \in carrier\ rel$ **and** $a3{:}y \in carrier\ rel$
then obtain *t* **where** $f1{:}t \in S$ **and** $f2{:}t \otimes ((snd\ y \otimes fst\ x) \ominus (snd\ x \otimes fst\ y)) = \mathbf{0}$
using *rel-def*
by *fastforce*
then have $(snd\ x \otimes fst\ y) \ominus (snd\ y \otimes fst\ x) = \ominus ((snd\ y \otimes fst\ x) \ominus (snd\ x \otimes fst\ y))$
using *abelian-group.minus-add abelian-group.minus-minus*

by (*smt a2 a3 a-minus-def abelian-group.a-inv-closed add.inv-mult-group is-abelian-group*
mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *prod.collapse*
rel-def rev-subsetD subset)
then have $t \otimes ((snd\ x \otimes fst\ y) \ominus (snd\ y \otimes fst\ x)) = \mathbf{0}$
using *minus-zero r-minus f2*
by (*smt a2 a3 f1 mem-Sigma-iff minus-closed partial-object.select-convs*(*1*) *prod.collapse*
rel-def semiring-simprules(*3*) *rev-subsetD subset*)
thus $y \mathrel{.=_{rel}} x$
using *f1 rel-def*
by *auto*
qed
show $\bigwedge x\ y\ z.$
$x \mathrel{.=_{rel}} y \Longrightarrow y \mathrel{.=_{rel}} z \Longrightarrow x \in carrier\ rel \Longrightarrow y \in carrier\ rel \Longrightarrow z \in carrier\ rel \Longrightarrow x \mathrel{.=_{rel}} z$
proof−
fix $x\ y\ z$
assume $a1{:}x \mathrel{.=_{rel}} y$ **and** $a2{:}y \mathrel{.=_{rel}} z$ **and** $a3{:}x \in carrier\ rel$ **and** $a4{:}y \in carrier\ rel$
and $a5{:}z \in carrier\ rel$
then obtain t **where** $f1{:}t \in S$ **and** $f2{:}t \otimes ((snd\ y \otimes fst\ x) \ominus (snd\ x \otimes fst\ y)) = \mathbf{0}$
using *rel-def*
by *fastforce*
then obtain t' **where** $f3{:}t' \in S$ **and** $f4{:}t' \otimes ((snd\ z \otimes fst\ y) \ominus (snd\ y \otimes fst\ z)) = \mathbf{0}$
using *rel-def a2*
by *fastforce*
then have $t \otimes (snd\ y \otimes fst\ x) \ominus t \otimes (snd\ x \otimes fst\ y) = \mathbf{0}$
using *f1 subset r-distr f2*
by (*smt a3 a4 a-minus-def abelian-group.a-inv-closed is-abelian-group mem-Sigma-iff*
monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *prod.collapse r-minus rel-def*
subset-iff)
then have $t' \otimes (t \otimes (snd\ y \otimes fst\ x)) \ominus t' \otimes (t \otimes (snd\ x \otimes fst\ y)) = \mathbf{0}$
using *f3 subset r-distr*
by (*smt a3 a4 a-minus-def f1 is-abelian-group mem-Sigma-iff minus-to-eq*
partial-object.select-convs(*1*) *prod.collapse r-neg rel-def semiring-simprules*(*3*) *subset-iff*)
then have $f5{:}snd\ z \otimes (t' \otimes (t \otimes (snd\ y \otimes fst\ x))) \ominus snd\ z \otimes (t' \otimes (t \otimes (snd\ x \otimes fst\ y))) = \mathbf{0}$
using *a5 rel-def r-distr*
by (*smt a3 a4 a-minus-def f1 f3 is-abelian-group mem-Sigma-iff minus-to-eq monoid.m-closed*
monoid-axioms partial-object.select-convs(*1*) *prod.collapse r-neg subset subset-iff*)

have $t' \otimes (snd\ z \otimes fst\ y) \ominus t' \otimes (snd\ y \otimes fst\ z) = \mathbf{0}$
using *f3 f4 subset r-distr*
by (*smt a4 a5 a-minus-def abelian-group.a-inv-closed is-abelian-group mem-Sigma-iff*

monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *prod.collapse r-minus rel-def*
rev-subsetD)
then have $t \otimes (t' \otimes (snd\ z \otimes fst\ y)) \ominus t \otimes (t' \otimes (snd\ y \otimes fst\ z)) = \mathbf{0}$
using *f1 subset r-distr*
by (*smt a4 a5 a-minus-def f3 is-abelian-group mem-Sigma-iff minus-to-eq monoid.m-closed*
monoid-axioms partial-object.select-convs(*1*) *prod.collapse r-neg rel-def subset-iff*)
then have *f6*:$snd\ x \otimes (t \otimes (t' \otimes (snd\ z \otimes fst\ y))) \ominus snd\ x \otimes (t \otimes (t' \otimes (snd\ y \otimes fst\ z))) = \mathbf{0}$
using *a3 rel-def r-distr*
by (*smt a4 a5 a-minus-def f1 f3 is-abelian-group mem-Sigma-iff minus-to-eq monoid.m-closed*
monoid-axioms partial-object.select-convs(*1*) *prod.collapse r-neg subset subset-iff*)
have $snd\ z \otimes (t' \otimes (t \otimes (snd\ x \otimes fst\ y))) = snd\ x \otimes (t \otimes (t' \otimes (snd\ z \otimes fst\ y)))$
using *comm-monoid-axioms-def*[*of R*] *f1 f3 subset a3 a4 a5 m-assoc*
by (*smt m-lcomm mem-Sigma-iff partial-object.select-convs*(*1*) *partial-object-ext-def rel-def*
semiring-simprules(*3*) *rev-subsetD surjective-pairing*)
then have $snd\ z \otimes (t' \otimes (t \otimes (snd\ y \otimes fst\ x))) \ominus snd\ z \otimes (t' \otimes (t \otimes (snd\ x \otimes fst\ y))) \oplus$
$snd\ x \otimes (t \otimes (t' \otimes (snd\ z \otimes fst\ y))) \ominus snd\ x \otimes (t \otimes (t' \otimes (snd\ y \otimes fst\ z)))$
$=$
$snd\ z \otimes (t' \otimes (t \otimes (snd\ y \otimes fst\ x))) \ominus snd\ x \otimes (t \otimes (t' \otimes (snd\ y \otimes fst\ z)))$
using *add.l-inv*
by (*smt a3 a4 a5 f1 f3 f5 is-abelian-group local.semiring-axioms mem-Sigma-iff minus-to-eq*
monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *prod.collapse rel-def*
semiring.semiring-simprules(*6*) *subset subset-iff*)
then have *f7*:$snd\ z \otimes (t' \otimes (t \otimes (snd\ y \otimes fst\ x))) \ominus snd\ x \otimes (t \otimes (t' \otimes (snd\ y \otimes fst\ z))) = \mathbf{0}$
using *f5 f6*
by (*smt* ‹$snd\ z \otimes (t' \otimes (t \otimes (snd\ x \otimes fst\ y))) = snd\ x \otimes (t \otimes (t' \otimes (snd\ z \otimes fst\ y)))$›
‹$t' \otimes (snd\ z \otimes fst\ y) \ominus t' \otimes (snd\ y \otimes fst\ z) = \mathbf{0}$› *a4 a5 f3 is-abelian-group mem-Sigma-iff*
minus-to-eq partial-object.select-convs(*1*) *prod.collapse rel-def semiring-simprules*(*3*)
subset subset-iff)
moreover have $(t \otimes t' \otimes snd\ y) \otimes ((snd\ z \otimes fst\ x) \ominus (snd\ x \otimes fst\ z)) = ((t \otimes t' \otimes snd\ y) \otimes (snd\ z \otimes fst\ x)) \ominus ((t \otimes t' \otimes snd\ y) \otimes (snd\ x \otimes fst\ z))$

using *r-distr f1 f3 subset a3 a4 a5 rel-def a-minus-def r-minus*
by (*smt SigmaE abelian-group.a-inv-closed is-abelian-group monoid.m-closed monoid-axioms*
partial-object.select-convs(*1*) *prod.sel*(*1*) *prod.sel*(*2*) *subset-iff*)
moreover have *f8*:$(t \otimes t' \otimes snd\ y) \otimes (snd\ z \otimes fst\ x) = snd\ z \otimes (t' \otimes (t \otimes (snd\ y \otimes fst\ x)))$
using *m-assoc comm-monoid-axioms-def*[*of R*] *f1 f3 subset a3 a4 a5 rel-def rev-subsetD*
by (*smt SigmaE local.semiring-axioms m-lcomm partial-object.select-convs*(*1*) *prod.sel*(*1*)
prod.sel(*2*) *semiring.semiring-simprules*(*3*))
moreover have *f9*:$(t \otimes t' \otimes snd\ y) \otimes (snd\ x \otimes fst\ z) = snd\ x \otimes (t \otimes (t' \otimes (snd\ y \otimes fst\ z)))$
using *m-assoc comm-monoid-axioms-def*[*of R*] *f1 f3 subset a3 a4 a5 rel-def rev-subsetD*
by (*smt SigmaE m-comm monoid.m-closed monoid-axioms partial-object.select-convs*(*1*) *prod.sel*(*1*)
prod.sel(*2*))
then have *f10*:$(t \otimes t' \otimes snd\ y) \otimes (snd\ z \otimes fst\ x) \ominus (t \otimes t' \otimes snd\ y) \otimes (snd\ x \otimes fst\ z) = \mathbf{0}$
using *f7 f8 f9*
by *simp*
moreover have $t \otimes t' \otimes snd\ y \in S$
using *f1 f3 a4 rel-def m-closed*
by (*simp add*: *mem-Times-iff*)
then have $(t \otimes t' \otimes snd\ y) \otimes (snd\ z \otimes fst\ x \ominus snd\ x \otimes fst\ z) = \mathbf{0}$
using *r-distr subset rev-subsetD f10 calculation*(*2*)
by *auto*
thus $x \mathrel{.=_{rel}} z$
using *rel-def* ‹$t \otimes t' \otimes snd\ y \in S$›
by *auto*
qed
qed

definition *eq-class-of-rng-of-frac*:: - ⇒ ′*a* ⇒ ′*b* ⇒ -*set* (**infix** ‹|ı› *10*)
where $r \mid_{rel} s \equiv \{(r', s') \in carrier\ rel.\ (r, s) \mathrel{.=_{rel}} (r', s')\}$

lemma *class-of-to-rel*:
shows $class\text{-}of_{rel}\ (r, s) = (r \mid_{rel} s)$
using *eq-class-of-def*[*of rel*] *eq-class-of-rng-of-frac-def*[*of rel*]
by *auto*

lemma (**in** *eq-obj-rng-of-frac*) *zero-in-mult-submonoid*:
assumes $\mathbf{0} \in S$ **and** $(r, s) \in carrier\ rel$ **and** $(r', s') \in carrier\ rel$
shows $(r \mid_{rel} s) = (r' \mid_{rel} s')$
proof
show $(r \mid_{rel} s) \subseteq (r' \mid_{rel} s')$
proof
fix x

```
    assume a1:x ∈ (r |rel s)
    have 0 ⊗ (s′ ⊗ fst x ⊖ snd x ⊗ r′) = 0
      using l-zero subset rel-def a1 eq-class-of-rng-of-frac-def
     by (smt abelian-group.minus-closed assms(3) is-abelian-group l-null mem-Collect-eq
mem-Sigma-iff
         monoid.m-closed monoid-axioms old.prod.case partial-object.select-convs(1)
subset-iff surjective-pairing)
    thus x ∈ (r′ |rel s′)
      using assms(1) assms(3) rel-def eq-class-of-rng-of-frac-def
     by (smt SigmaE a1 eq-object.select-convs(1) l-null mem-Collect-eq minus-closed
old.prod.case
           partial-object.select-convs(1) prod.collapse semiring-simprules(3) subset
subset-iff)
  qed
  show (r′ |rel s′) ⊆ (r |rel s)
  proof
    fix x
    assume a1:x ∈ (r′ |rel s′)
    have 0 ⊗ (s ⊗ fst x ⊖ snd x ⊗ r) = 0
      using l-zero subset rel-def a1 eq-class-of-rng-of-frac-def
        by (metis (no-types, lifting) BNF-Def.Collect-case-prodD assms(2) l-null
mem-Sigma-iff
        minus-closed partial-object.select-convs(1) semiring-simprules(3) rev-subsetD)
    thus x ∈ (r |rel s)
      using assms(1) assms(2) rel-def eq-class-of-rng-of-frac-def
     by (smt SigmaE a1 eq-object.select-convs(1) l-null mem-Collect-eq minus-closed
old.prod.case
           partial-object.select-convs(1) prod.collapse semiring-simprules(3) subset
subset-iff)
  qed
qed

definition set-eq-class-of-rng-of-frac:: - ⇒ -set (‹set′-class′-of1›)
  where set-class-ofrel ≡ {(r |rel s)| r s. (r, s) ∈ carrier rel}


lemma elem-eq-class:
  assumes equivalence S and x ∈ carrier S and y ∈ carrier S and x .=S y
  shows class-ofS x = class-ofS y
proof
  show class-ofS x ⊆ class-ofS y
  proof
    fix z
    assume z ∈ class-ofS x
    then have y .=S z
     using assms eq-class-of-def[of S x] equivalence.sym[of S x y] equivalence.trans
      by (metis (mono-tags, lifting) mem-Collect-eq)
    thus z ∈ class-ofS y
      using ‹z ∈ class-ofS x›
```

by (*simp add*: *eq-class-of-def*)
qed
show *class-of*$_S$ *y* ⊆ *class-of*$_S$ *x*
proof
fix *z*
assume *z* ∈ *class-of*$_S$ *y*
then have *x* .=$_S$ *z*
using *assms eq-class-of-def equivalence.trans*
by (*metis* (*mono-tags*, *lifting*) *mem-Collect-eq*)
thus *z* ∈ *class-of*$_S$ *x*
using ‹*z* ∈ *class-of*$_S$ *y*›
by (*simp add*: *eq-class-of-def*)
qed
qed

lemma (**in** *abelian-group*) *four-elem-comm*:
assumes *a* ∈ *carrier G* **and** *b* ∈ *carrier G* **and** *c* ∈ *carrier G* **and** *d* ∈ *carrier G*
shows *a* ⊖ *c* ⊕ *b* ⊖ *d* = *a* ⊕ *b* ⊖ *c* ⊖ *d*
using *assms a-assoc a-comm*
by (*simp add*: *a-minus-def*)

lemma (**in** *abelian-monoid*) *right-add-eq*:
assumes *a* = *b*
shows *c* ⊕ *a* = *c* ⊕ *b*
using *assms*
by *simp*

lemma (**in** *abelian-monoid*) *right-minus-eq*:
assumes *a* = *b*
shows *c* ⊖ *a* = *c* ⊖ *b*
by (*simp add*: *assms*)

lemma (**in** *abelian-group*) *inv-add*:
assumes *a* ∈ *carrier G* **and** *b* ∈ *carrier G*
shows ⊖ (*a* ⊕ *b*) = ⊖ *a* ⊖ *b*
using *assms minus-add*
by (*simp add*: *a-minus-def*)

lemma (**in** *abelian-group*) *right-inv-add*:
assumes *a* ∈ *carrier G* **and** *b* ∈ *carrier G* **and** *c* ∈ *carrier G*
shows *c* ⊖ *a* ⊖ *b* = *c* ⊖ (*a* ⊕ *b*)
using *assms*
by (*simp add*: *a-minus-def add.m-assoc local.minus-add*)

context *eq-obj-rng-of-frac*
begin

definition *carrier-rng-of-frac*:: - *partial-object*

where *carrier-rng-of-frac* ≡ (|*carrier* = *set-class-of*$_{rel}$|)

definition *mult-rng-of-frac*:: [-*set*, -*set*] ⇒ -*set*
where *mult-rng-of-frac X Y* ≡
let x' = (*SOME x*. *x* ∈ *X*) *in*
let y' = (*SOME y*. *y* ∈ *Y*) *in*
(*fst x'* ⊗ *fst y'*)$|_{rel}$ (*snd x'* ⊗ *snd y'*)

definition *rec-monoid-rng-of-frac*:: - *monoid*
where *rec-monoid-rng-of-frac* ≡ (|*carrier* = *set-class-of*$_{rel}$, *mult* = *mult-rng-of-frac*, *one* = (**1**$|_{rel}$ **1**)|)

lemma *member-class-to-carrier*:
assumes *x* ∈ (*r* $|_{rel}$ *s*) **and** *y* ∈ (*r'* $|_{rel}$ *s'*)
shows (*fst x* ⊗ *fst y*, *snd x* ⊗ *snd y*) ∈ *carrier rel*
using *assms rel-def eq-class-of-rng-of-frac-def*
by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD m-closed mem-Sigma-iff*

partial-object.select-convs(*1*) *semiring-simprules*(*3*))

lemma *member-class-to-member-class*:
assumes *x* ∈ (*r* $|_{rel}$ *s*) **and** *y* ∈ (*r'* $|_{rel}$ *s'*)
shows (*fst x* ⊗ *fst y* $|_{rel}$ *snd x* ⊗ *snd y*) ∈ *set-class-of*$_{rel}$
using *assms member-class-to-carrier*[*of x r s y r' s'*] *set-eq-class-of-rng-of-frac-def*[*of rel*]
eq-class-of-rng-of-frac-def
by *auto*

lemma *closed-mult-rng-of-frac* :
assumes (*r*, *s*) ∈ *carrier rel* **and** (*t*, *u*) ∈ *carrier rel*
shows (*r* $|_{rel}$ *s*) $\otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}$ (*t* $|_{rel}$ *u*) ∈ *set-class-of*$_{rel}$
proof−
have (*r*, *s*) .=$_{rel}$ (*r*, *s*)
using *assms*(*1*) *equiv-obj-rng-of-frac equivalence-def*[*of rel*]
by *blast*
then have (*r*, *s*) ∈ (*r* $|_{rel}$ *s*)
using *assms*(*1*)
by (*simp add*: *eq-class-of-rng-of-frac-def*)
then have *f1*:∃ *x*. *x* ∈ (*r* $|_{rel}$ *s*)
by *auto*
have *f2*:∃ *y*. *y*∈ (*t* $|_{rel}$ *u*)
using *assms*(*2*) *equiv-obj-rng-of-frac equivalence.refl eq-class-of-rng-of-frac-def*
by *fastforce*
show (*r* $|_{rel}$ *s*) $\otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}$ (*t* $|_{rel}$ *u*) ∈ *set-class-of*$_{rel}$
using *f1 f2 rec-monoid-rng-of-frac-def mult-rng-of-frac-def*[*of* (*r* $|_{rel}$ *s*) (*t* $|_{rel}$ *u*)]
set-eq-class-of-rng-of-frac-def[*of rel*] *member-class-to-member-class*[*of x' r s y' t u*]
by (*metis* (*mono-tags*, *lifting*) *mem-Collect-eq member-class-to-carrier monoid.select-convs*(*1*)

someI-ex)
qed

lemma *non-empty-class*:
assumes $(r, s) \in$ *carrier rel*
shows $(r\ |_{rel}\ s) \neq \{\}$
using *assms eq-class-of-rng-of-frac-def equiv-obj-rng-of-frac equivalence.refl*
by *fastforce*

lemma *mult-rng-of-frac-fundamental-lemma*:
assumes $(r, s) \in$ *carrier rel* **and** $(r', s') \in$ *carrier rel*
shows $(r\ |_{rel}\ s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (r \otimes r'\ |_{rel}\ s \otimes s')$
proof−
have *f1*:$(r\ |_{rel}\ s) \neq \{\}$
using *assms*(*1*) *non-empty-class*
by *auto*
have $(r'\ |_{rel}\ s') \neq \{\}$
using *assms*(*2*) *non-empty-class*
by *auto*
then have $\exists x \in (r\ |_{rel}\ s).\ \exists x' \in (r'\ |_{rel}\ s').\ (r\ |_{rel}\ s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') =$
$(fst\ x \otimes fst\ x'\ |_{rel}\ snd\ x \otimes snd\ x')$
using *f1 rec-monoid-rng-of-frac-def*
by (*metis monoid.select-convs*(*1*) *mult-rng-of-frac-def some-in-eq*)
then obtain x **and** x' **where** *f2*:$x \in (r\ |_{rel}\ s)$ **and** *f3*:$x' \in (r'\ |_{rel}\ s')$
and $(r\ |_{rel}\ s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (fst\ x \otimes fst\ x'\ |_{rel}\ snd\ x \otimes snd\ x')$
by *blast*
then have $(r, s)\ .=_{rel} (fst\ x, snd\ x)$
using *rel-def*
by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def*)
then obtain t **where** *f4*:$t \in S$ **and** *f5*:$t \otimes ((snd\ x \otimes r) \ominus (s \otimes fst\ x)) = \mathbf{0}$
using *rel-def*
by *auto*
have $(r', s')\ .=_{rel} (fst\ x', snd\ x')$
using *rel-def f3*
by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def*)
then obtain t' **where** *f6*:$t' \in S$ **and** *f7*:$t' \otimes (snd\ x' \otimes r' \ominus s' \otimes fst\ x') = \mathbf{0}$
using *rel-def*
by *auto*
have *f8*:$t \in$ *carrier R*
using *f4 subset rev-subsetD*
by *auto*
have *f9*:$snd\ x \otimes r \in$ *carrier R*
using *subset rev-subsetD f2 assms*(*1*)
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff*
partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*))

have *f10*:$\ominus$ (*s* $\otimes$ *fst x*) $\in$ *carrier R*
using *assms*(*1*) *subset rev-subsetD f2*
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD abelian-group.a-inv-closed*

eq-class-of-rng-of-frac-def is-abelian-group mem-Sigma-iff monoid.m-closed monoid-axioms
partial-object.select-convs(*1*) *rel-def*)
then have $t \otimes (snd\ x \otimes r) \ominus t \otimes (s \otimes fst\ x) = \mathbf{0}$
using *f8 f9 f10 f5 r-distr*[*of snd x* $\otimes$ *r* $\ominus$ (*s* $\otimes$ *fst x*) *t*] *a-minus-def r-minus*[*of t s* $\otimes$ *fst x*]
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *eq-class-of-rng-of-frac-def f2 mem-Sigma-iff*
partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have *f11*:$t \otimes (snd\ x \otimes r) = t \otimes (s \otimes fst\ x)$
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *eq-class-of-rng-of-frac-def f2 f8 is-abelian-group*
mem-Sigma-iff minus-to-eq monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def subset subset-iff*)
have *f12*:$t' \in$ *carrier R*
using *f6 subset rev-subsetD*
by *auto*
have *f13*:$snd\ x' \otimes r' \in$ *carrier R*
using *assms*(*2*) *f3 subset rev-subsetD*
by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def*

mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def*)
have *f14*:$\ominus (s' \otimes fst\ x') \in$ *carrier R*
using *assms*(*2*) *f3 subset rev-subsetD*
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD abelian-group.a-inv-closed*

eq-class-of-rng-of-frac-def is-abelian-group mem-Sigma-iff monoid.m-closed monoid-axioms
partial-object.select-convs(*1*) *rel-def*)
then have $t' \otimes (snd\ x' \otimes r') \ominus t' \otimes (s' \otimes fst\ x') = \mathbf{0}$
using *f12 f13 f14 f7 r-distr*[*of snd x'* $\otimes$ *r'* $\ominus$ (*s'* $\otimes$ *fst x'*) *t'*] *a-minus-def*
r-minus[*of t' s'* $\otimes$ *fst x'*]
by (*smt BNF-Def.Collect-case-prodD assms*(*2*) *eq-class-of-rng-of-frac-def f3 mem-Sigma-iff*
partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have *f15*:$t' \otimes (snd\ x' \otimes r') = t' \otimes (s' \otimes fst\ x')$
by (*smt BNF-Def.Collect-case-prodD assms*(*2*) *eq-class-of-rng-of-frac-def f3 f12 is-abelian-group*
mem-Sigma-iff minus-to-eq monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def subset subset-iff*)
have $t' \otimes t \in S$
using *f4 f6 m-closed*
by *auto*
then have *f16*:$t' \otimes t \in$ *carrier R*

using *subset rev-subsetD*
by *auto*
have *f17*:(*snd x* ⊗ *snd x*′) ⊗ (*r* ⊗ *r*′) ∈ *carrier R*
using *assms f2 f3*
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff*
monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def subset subset-iff*)
have *f18*:(*s* ⊗ *s*′) ⊗ (*fst x* ⊗ *fst x*′) ∈ *carrier R*
using *assms f2 f3*
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff*
monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def subset subset-iff*)
then have *f19*:(*t*′ ⊗ *t*) ⊗ ((*snd x* ⊗ *snd x*′) ⊗ (*r* ⊗ *r*′) ⊖ (*s* ⊗ *s*′) ⊗ (*fst x* ⊗ *fst x*′)) =
((*t*′ ⊗ *t*) ⊗ (*snd x* ⊗ *snd x*′)) ⊗ (*r* ⊗ *r*′) ⊖ (*t*′ ⊗ *t*) ⊗ ((*s* ⊗ *s*′) ⊗ (*fst x* ⊗ *fst x*′))
using *f16 f17 f18 r-distr m-assoc r-minus a-minus-def*
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *assms*(*2*) *eq-class-of-rng-of-frac-def f14 f2 f3*
m-comm mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def*
subset subset-iff)
then have *f20*:(*t*′ ⊗ *t*) ⊗ (*snd x* ⊗ *snd x*′) ⊗ (*r* ⊗ *r*′) = (*t*′ ⊗ *t*) ⊗ (*snd x* ⊗ *r* ⊗ *snd x*′ ⊗ *r*′)
using *m-assoc m-comm f16 assms rel-def f2 f3*
by (*smt BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff*

partial-object.select-convs(*1*) *semiring-simprules*(*3*) *subset subset-iff*)
then have ((*t*′ ⊗ *t*) ⊗ (*snd x* ⊗ *snd x*′)) ⊗ (*r* ⊗ *r*′) = *t*′ ⊗ ((*t* ⊗ *snd x* ⊗ *r*) ⊗ *snd x*′ ⊗ *r*′)
using *m-assoc assms f2 f3 rel-def f8 f12*
by (*smt BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff monoid.m-closed*
monoid-axioms partial-object.select-convs(*1*) *subset subset-iff*)
then have *f21*:((*t*′ ⊗ *t*) ⊗ (*snd x* ⊗ *snd x*′)) ⊗ (*r* ⊗ *r*′) = *t*′ ⊗ (*t* ⊗ *s* ⊗ *fst x*) ⊗ *snd x*′ ⊗ *r*′
using *f11 m-assoc*
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *assms*(*2*) *eq-class-of-rng-of-frac-def f12 f2 f3 f8*
mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def subset subset-iff*)
moreover have (*t*′ ⊗ *t*) ⊗ ((*s* ⊗ *s*′) ⊗ (*fst x* ⊗ *fst x*′)) = (*t*′ ⊗ *s*′ ⊗ *fst x*′) ⊗ *t* ⊗ *s* ⊗ *fst x*
using *assms f2 f3 f8 f12 m-assoc m-comm rel-def*
by (*smt BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff monoid.m-closed*
monoid-axioms partial-object.select-convs(*1*) *subset subset-iff*)

then have $(t' \otimes t) \otimes ((s \otimes s') \otimes (fst\ x \otimes fst\ x')) = (t' \otimes snd\ x' \otimes r') \otimes t \otimes s \otimes fst\ x$
using *f15 m-assoc*
by (*smt BNF-Def.Collect-case-prodD assms*(*2*) *eq-class-of-rng-of-frac-def f12 f3 mem-Sigma-iff*
partial-object.select-convs(*1*) *rel-def subset subset-iff*)
then have *f22*:$(t' \otimes t) \otimes ((s \otimes s') \otimes (fst\ x \otimes fst\ x')) = t' \otimes ((t \otimes snd\ x \otimes r) \otimes snd\ x' \otimes r')$
using *m-assoc m-comm assms*
by (*smt BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def f12 f2 f21 f3 f8 mem-Sigma-iff*
partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have *f23*:$(t' \otimes t) \otimes ((snd\ x \otimes snd\ x') \otimes (r \otimes r') \ominus (s \otimes s') \otimes (fst\ x \otimes fst\ x')) = \mathbf{0}$
using *f19 f21 f22*
by (*metis* ‹$t' \otimes t \otimes (snd\ x \otimes snd\ x') \otimes (r \otimes r') = t' \otimes (t \otimes snd\ x \otimes r \otimes snd\ x' \otimes r')$›
a-minus-def f16 f18 r-neg semiring-simprules(*3*))
have *f24*:$(r \otimes r', s \otimes s') \in carrier\ rel$
using *assms rel-def*
by *auto*
have *f25*: $(fst\ x \otimes fst\ x', snd\ x \otimes snd\ x') \in carrier\ rel$
using *f2 f3 member-class-to-carrier*
by *auto*
then have $(r \otimes r', s \otimes s') .=_{rel} (fst\ x \otimes fst\ x', snd\ x \otimes snd\ x')$
using *f23 f24 rel-def* ‹$t' \otimes t \in S$›
by *auto*
then have $class\text{-}of_{rel}\ (r \otimes r', s \otimes s') = class\text{-}of_{rel}\ (fst\ x \otimes fst\ x', snd\ x \otimes snd\ x')$
using *f24 f25 equiv-obj-rng-of-frac elem-eq-class*[*of rel* $(r \otimes r', s \otimes s')$ $(fst\ x \otimes fst\ x', snd\ x \otimes snd\ x')$]
eq-class-of-rng-of-frac-def
by *auto*
then have $(r \otimes r' |_{rel}\ s \otimes s') = (fst\ x \otimes fst\ x' |_{rel}\ snd\ x \otimes snd\ x')$
using *class-of-to-rel*[*of rel*]
by *auto*
thus *?thesis*
using ‹$(r |_{rel}\ s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r' |_{rel}\ s') = (fst\ x \otimes fst\ x' |_{rel}\ snd\ x \otimes snd\ x')$›
trans sym
by *auto*
qed

lemma *member-class-to-assoc*:
assumes $x \in (r |_{rel}\ s)$ **and** $y \in (t |_{rel}\ u)$ **and** $z \in (v |_{rel}\ w)$
shows $((fst\ x \otimes fst\ y) \otimes fst\ z |_{rel}\ (snd\ x \otimes snd\ y) \otimes snd\ z) = (fst\ x \otimes (fst\ y \otimes fst\ z) |_{rel}\ snd\ x \otimes (snd\ y \otimes snd\ z))$
using *assms m-assoc subset rel-def rev-subsetD*
by (*smt BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff*

partial-object.select-convs(*1*))

lemma *assoc-mult-rng-of-frac*:
 assumes $(r, s) \in$ *carrier rel* **and** $(t, u) \in$ *carrier rel* **and** $(v, w) \in$ *carrier rel*
 shows $((r \mid_{rel} s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (t \mid_{rel} u)) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (v \mid_{rel}$ $w) =$
 $(r \mid_{rel} s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} ((t \mid_{rel} u) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (v \mid_{rel}$ $w))$
proof−
 have $((r \otimes t) \otimes v, (s \otimes u) \otimes w) = (r \otimes (t \otimes v), s \otimes (u \otimes w))$
 using *assms m-assoc*
 by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*) *rel-def rev-subsetD subset*)
 then have *f1*:$((r \otimes t) \otimes v \mid_{rel} (s \otimes u) \otimes w) = (r \otimes (t \otimes v) \mid_{rel} s \otimes (u \otimes w))$
 by *simp*
 have *f2*:$((r \mid_{rel} s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (t \mid_{rel} u)) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (v$ $\mid_{rel} w) =$
 $((r \otimes t) \otimes v \mid_{rel} (s \otimes u) \otimes w)$
 using *assms mult-rng-of-frac-fundamental-lemma rel-def*
 by *auto*
 have *f3*:$(r \mid_{rel} s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} ((t \mid_{rel} u) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (v \mid_{rel}$ $w)) =$
 $(r \otimes (t \otimes v) \mid_{rel} s \otimes (u \otimes w))$
 using *assms mult-rng-of-frac-fundamental-lemma rel-def*
 by *auto*
 thus *?thesis*
 using *f1 f2 f3*
 by *simp*
qed

lemma *left-unit-mult-rng-of-frac*:
 assumes $(r, s) \in$ *carrier rel*
 shows $\mathbf{1}_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r \mid_{rel} s) = (r \mid_{rel} s)$
 using *assms subset rev-subsetD rec-monoid-rng-of-frac-def mult-rng-of-frac-fundamental-lemma*[*of* **1** **1** *r s*]
 l-one[*of r*] *l-one*[*of s*] *rel-def*
 by *auto*

lemma *right-unit-mult-rng-of-frac*:
 assumes $(r, s) \in$ *carrier rel*
 shows $(r \mid_{rel} s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} \mathbf{1}_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} = (r \mid_{rel} s)$
 using *assms subset rev-subsetD rec-monoid-rng-of-frac-def mult-rng-of-frac-fundamental-lemma*[*of* *r s* **1** **1**]
 r-one[*of r*] *r-one*[*of s*] *rel-def*
 by *auto*

lemma *monoid-rng-of-frac*:
 shows *monoid* (*rec-monoid-rng-of-frac*)
proof

show $\bigwedge x\ y.\ x \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow$
$y \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow x \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}\ y \in$ *carrier rec-monoid-rng-of-frac*
using *rec-monoid-rng-of-frac-def closed-mult-rng-of-frac*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *set-eq-class-of-rng-of-frac-def*)
show $\bigwedge x\ y\ z.\ x \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow$
$y \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow$
$z \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow$
$x \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}\ y \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}\ z =$
$x \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}\ (y \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}\ z)$
using *assoc-mult-rng-of-frac*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *rec-monoid-rng-of-frac-def*

set-eq-class-of-rng-of-frac-def)
show $\mathbf{1}_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} \in$ *carrier rec-monoid-rng-of-frac*
using *rec-monoid-rng-of-frac-def rel-def set-eq-class-of-rng-of-frac-def*
by *fastforce*
show $\bigwedge x.\ x \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow \mathbf{1}_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}$ $x = x$
using *left-unit-mult-rng-of-frac*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *rec-monoid-rng-of-frac-def set-eq-class-of-rng-of-frac-def*)
show $\bigwedge x.\ x \in$ *carrier rec-monoid-rng-of-frac* $\Longrightarrow x \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} \mathbf{1}_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}$ $= x$
using *right-unit-mult-rng-of-frac*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *rec-monoid-rng-of-frac-def set-eq-class-of-rng-of-frac-def*)
qed

lemma *comm-mult-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel* **and** $(r', s') \in$ *carrier rel*
shows $(r\ |_{rel}\ s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (r'\ |_{rel}\ s') \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac}$ $(r\ |_{rel}\ s)$
proof−
have *f1*:$(r\ |_{rel}\ s) \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (r \otimes r'\ |_{rel}\ s \otimes s')$
using *assms mult-rng-of-frac-fundamental-lemma*
by *simp*
have *f2*:$(r'\ |_{rel}\ s') \otimes_{rec\text{-}monoid\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) = (r' \otimes r\ |_{rel}\ s' \otimes s)$
using *assms mult-rng-of-frac-fundamental-lemma*
by *simp*
have *f3*:$r \otimes r' = r' \otimes r$
using *assms rel-def m-comm*
by *simp*
have *f4*:$s \otimes s' = s' \otimes s$
using *assms rel-def subset rev-subsetD m-comm*
by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*))
thus *?thesis*
using *f1 f2 f3 f4*
by *simp*

qed

lemma *comm-monoid-rng-of-frac*:
shows *comm-monoid* (*rec-monoid-rng-of-frac*)
using *comm-monoid-def Group.comm-monoid-axioms-def monoid-rng-of-frac comm-mult-rng-of-frac*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *rec-monoid-rng-of-frac-def set-eq-class-of-rng-of-frac-def*)

definition *add-rng-of-frac*:: [-*set*, -*set*] ⇒ -*set*
where *add-rng-of-frac X Y* ≡
let x' = (*SOME x. x* ∈ *X*) *in*
let y' = (*SOME y. y* ∈ *Y*) *in*
(*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y') $|_{rel}$ (*snd* x' ⊗ *snd* y')

definition *rec-rng-of-frac*:: - *ring*
where *rec-rng-of-frac* ≡
(|*carrier* = *set-class-of*$_{rel}$, *mult* = *mult-rng-of-frac*, *one* = ($\mathbf{1}|_{rel}$ **1**), *zero* = (**0** $|_{rel}$ **1**), *add* = *add-rng-of-frac* |)

lemma *add-rng-of-frac-fundamental-lemma*:
assumes (r, s) ∈ *carrier rel* **and** (r', s') ∈ *carrier rel*
shows (r $|_{rel}$ s) $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ (r' $|_{rel}$ s') = (s' ⊗ r ⊕ s ⊗ r' $|_{rel}$ s ⊗ s')
proof−
have ∃ x' ∈ (r $|_{rel}$ s). ∃ y' ∈ (r' $|_{rel}$ s'). (r $|_{rel}$ s) $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ (r' $|_{rel}$ s') = (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y' $|_{rel}$ *snd* x' ⊗ *snd* y')
using *assms rec-rng-of-frac-def add-rng-of-frac-def*[*of* (r $|_{rel}$ s) (r' $|_{rel}$ s')]
by (*metis non-empty-class ring-record-simps*(*12*) *some-in-eq*)
then obtain x' **and** y' **where** *f1*:x' ∈ (r $|_{rel}$ s) **and** *f2*:y' ∈ (r' $|_{rel}$ s') **and** *f3*:(r $|_{rel}$ s) $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ (r' $|_{rel}$ s') = (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y' $|_{rel}$ *snd* x' ⊗ *snd* y')
by *auto*
then have (r, s) $.=_{rel}$ x'
using *f1 rel-def eq-class-of-rng-of-frac-def*[*of rel r s*]
by *auto*
then obtain t **where** *f4*:t ∈ S **and** *f5*:t ⊗ (*snd* x' ⊗ r ⊖ s ⊗ *fst* x') = **0**
using *rel-def*
by *auto*
have (r', s') $.=_{rel}$ y'
using *f2 rel-def eq-class-of-rng-of-frac-def*[*of rel* r' s']
by *auto*
then obtain t' **where** *f6*:t' ∈ S **and** *f7*:t' ⊗ (*snd* y' ⊗ r' ⊖ s' ⊗ *fst* y') = **0**
using *rel-def*
by *auto*
then have *f8*:t ⊗ t' ∈ S
using *m-closed f4 f6*
by *simp*
then have (s' ⊗ r ⊕ s ⊗ r', s ⊗ s') $.=_{rel}$ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y', *snd* x' ⊗ *snd* y')
proof−

have *f9*: $t' \otimes s' \otimes$ *snd* $y' \in$ *carrier R*
  using *f6 assms*(*2*) *f2 subset rev-subsetD eq-class-of-rng-of-frac-def rel-def*
  by *fastforce*
have *f10*:*snd* $x' \otimes r \in$ *carrier R*
  using *assms*(*1*) *f1 rel-def subset rev-subsetD*
 by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def*

  mem-Sigma-iff partial-object.select-convs(*1*) *semiring-simprules*(*3*))
have *f11*: $s \otimes$ *fst* $x' \in$ *carrier R*
  using *assms*(*1*) *subset rev-subsetD f1 rel-def*
 by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def*

  mem-Sigma-iff partial-object.select-convs(*1*) *semiring-simprules*(*3*))
have $t \otimes$ (*snd* $x' \otimes r \ominus s \otimes$ *fst* x') $= t \otimes$ (*snd* $x' \otimes r$) $\ominus t \otimes$ ($s \otimes$ *fst* x')
  using *f9 f10 f11 f4 subset rev-subsetD r-distr*[*of snd* $x' \otimes r\ s \otimes$ *fst* $x'\ t$]
a-minus-def
  r-minus[*of t s* $\otimes$ *fst* x']
  by (*smt add.inv-closed monoid.m-closed monoid-axioms r-distr*)
then have *f12*:($t' \otimes s' \otimes$ *snd* y') $\otimes$ ($t \otimes$ (*snd* $x' \otimes r \ominus s \otimes$ *fst* x')) =
 $t' \otimes s' \otimes$ *snd* $y' \otimes t \otimes$ (*snd* $x' \otimes r$) $\ominus$ ($t' \otimes s' \otimes$ *snd* $y' \otimes t \otimes$ ($s \otimes$ *fst* x'))
  using *f9 r-distr*[*of - -* $t' \otimes s' \otimes$ *snd* y'] *rel-def r-minus a-minus-def*
  by (*smt abelian-group.minus-to-eq f10 f11 f4 f5 is-abelian-group m-assoc*
monoid.m-closed
  monoid-axioms r-neg r-null subset subset-iff)
have *f13*:(*snd* $x' \otimes$ *snd* y') $\otimes$ ($s' \otimes r$) $\in$ *carrier R*
  using *assms f1 f2 subset rev-subsetD*
 by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def*

  mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*)
rel-def)
have *f14*:($s \otimes s'$) $\otimes$ (*snd* $y' \otimes$ *fst* x') $\in$ *carrier R*
  using *assms f1 f2 subset rev-subsetD*
 by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def*

  mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*)
rel-def)
then have ($t \otimes t'$) $\otimes$ ((*snd* $x' \otimes$ *snd* y') $\otimes$ ($s' \otimes r$) $\ominus$ ($s \otimes s'$) $\otimes$ (*snd* $y' \otimes$ *fst*
x')) =
 ($t \otimes t'$) $\otimes$ ((*snd* $x' \otimes$ *snd* y') $\otimes$ ($s' \otimes r$)) $\ominus$ ($t \otimes t'$) $\otimes$ (($s \otimes s'$) $\otimes$ (*snd* $y' \otimes$
fst x'))
  using *f13 f14 f8 subset rev-subsetD r-distr rel-def r-minus a-minus-def*
  by (*smt add.inv-closed semiring-simprules*(*3*))
have *f15*: $s \otimes s' \in$ *carrier R*
  using *assms rel-def subset rev-subsetD*
  by *auto*
have *f16*:*snd* $y' \otimes$ *fst* $x' \in$ *carrier R*
  using *f1 f2 rel-def subset rev-subsetD*[*of - S*] *monoid.m-closed*[*of R snd* y' *fst*
x']
 by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def*

mem-Sigma-iff monoid-axioms partial-object.select-convs(*1*))
have *f17*:$t \in$ *carrier R*
using *f4 subset rev-subsetD*
by *auto*
have *f18*:$t' \in$ *carrier R*
using *f6 subset rev-subsetD*
by *auto*
have *f19*:$s \in$ *carrier R*
using *assms*(*1*) *rel-def subset*
by *auto*
have *f20*:$s' \in$ *carrier R*
using *assms*(*2*) *rel-def subset*
by *auto*
have *f21*:*snd* $y' \in$ *carrier R*
using *f2 rel-def subset rev-subsetD*
by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def*

mem-Sigma-iff partial-object.select-convs(*1*))
have *f22*:*fst* $x' \in$ *carrier R*
using *f1 rel-def*
by (*metis* (*no-types*, *lifting*) *Product-Type.Collect-case-prodD eq-class-of-rng-of-frac-def mem-Sigma-iff*
partial-object.select-convs(*1*))
then have *f23*:$(t \otimes t') \otimes ((s \otimes s') \otimes (snd\ y' \otimes fst\ x')) = t' \otimes s' \otimes snd\ y' \otimes t \otimes (s \otimes fst\ x')$
using *f17 f18 f19 f20 f21 m-assoc m-comm*
by (*smt BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def f1 f4 f6 mem-Sigma-iff*

partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset-iff*)
have *f24*:$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r)) = t' \otimes s' \otimes snd\ y' \otimes t \otimes (snd\ x' \otimes r)$
using *f17 f18 f20 f21 m-assoc m-comm*
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *eq-class-of-rng-of-frac-def f1 f2 f4 f6*
mem-Sigma-iff partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have $(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r)) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ y' \otimes fst\ x'))=$
$(t' \otimes s' \otimes snd\ y' \otimes t \otimes (snd\ x' \otimes r)) \ominus (t' \otimes s' \otimes snd\ y' \otimes t \otimes (s \otimes fst\ x'))$
using *f23 f24*
by *simp*
then have *f25*:$(t' \otimes s' \otimes snd\ y') \otimes (t \otimes (snd\ x' \otimes r \ominus s \otimes fst\ x')) =$
$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r)) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ y' \otimes fst\ x'))$
using *f12*
by *simp*
have *f26*:$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r')) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ x' \otimes fst\ y')) =$

$t \otimes s \otimes snd\ x' \otimes t' \otimes (snd\ y' \otimes r') \ominus (t \otimes s \otimes snd\ x' \otimes t' \otimes (s' \otimes fst\ y'))$
by (*smt BNF-Def.Collect-case-prodD assms*(*2*) *eq-class-of-rng-of-frac-def f1 f17 f18 f19 f2*
m-assoc m-comm mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def subset subset-iff*)
have *f27*: $snd\ y' \otimes r' \in carrier\ R$
using *assms*(*2*) *f21 rel-def*
by *auto*
have *f28*: $s' \otimes fst\ y' \in carrier\ R$
using *f20 assms*(*2*)
by (*metis* (*no-types, lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def f2*
mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def*)
then have $t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y') = t' \otimes (snd\ y' \otimes r') \ominus t' \otimes (s' \otimes fst\ y')$
using *f18 f27 f28 r-minus*[*of* $t'\ s' \otimes fst\ y'$]
by (*simp add*: *a-minus-def r-distr*)
then have *f29*: $(t \otimes s \otimes snd\ x') \otimes (t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y')) =$
$(t \otimes s \otimes snd\ x') \otimes (t' \otimes (snd\ y' \otimes r') \ominus t' \otimes (s' \otimes fst\ y'))$
by *simp*
have $t \otimes s \otimes snd\ x' \in carrier\ R$
using *f17 f19 f1 subset assms*(*1*) *eq-class-of-rng-of-frac-def f4 rel-def*
by *fastforce*
then have *f30*: $(t \otimes s \otimes snd\ x') \otimes (t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y')) =$
$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r')) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ x' \otimes fst\ y'))$
using *f26 f29 r-distr*
by (*smt* ‹$t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y') = t' \otimes (snd\ y' \otimes r') \ominus t' \otimes (s' \otimes fst\ y')$›
a-minus-def abelian-group.minus-to-eq f18 f27 f28 f7 is-abelian-group m-assoc monoid.m-closed
monoid-axioms r-neg semiring-simprules(*15*))
then have *f31*: $((t' \otimes s' \otimes snd\ y') \otimes (t \otimes (snd\ x' \otimes r \ominus s \otimes fst\ x'))) \oplus ((t \otimes s \otimes snd\ x') \otimes (t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y')))$
$= ((t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r)) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ y' \otimes fst\ x'))) \oplus$
$((t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r')) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ x' \otimes fst\ y')))$
using *f25 f30*
by *simp*
have *f32*: $(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r)) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ y' \otimes fst\ x'))$
$= (t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x'))$
using *f17 f18 r-distr*
by (*simp add*: ‹$t \otimes t' \otimes (snd\ x' \otimes snd\ y' \otimes (s' \otimes r) \ominus s \otimes s' \otimes (snd\ y' \otimes fst\ x')) = t \otimes t' \otimes (snd\ x' \otimes snd\ y' \otimes (s' \otimes r)) \ominus t \otimes t' \otimes (s \otimes s' \otimes (snd\ y' \otimes fst\ x'))$›)
have *f33*: $(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r')) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes$

(*snd* x' ⊗ *fst* y')) =
(*t* ⊗ *t'*) ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (*s* ⊗ *r'*) ⊖ (*s* ⊗ *s'*) ⊗ (*snd* x' ⊗ *fst* y'))
using *r-distr*[*of* - - *t* ⊗ *t'*] *f17 f18 a-minus-def r-minus*
by (*smt BNF-Def.Collect-case-prodD abelian-group.a-inv-closed assms*(*1*) *assms*(*2*)
eq-class-of-rng-of-frac-def f1 f2 is-abelian-group mem-Sigma-iff partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
have *f34*:(*snd* x' ⊗ *snd* y') ⊗ (*s'* ⊗ *r* ⊕ *s* ⊗ *r'*) = (*snd* x' ⊗ *snd* y') ⊗ (*s'* ⊗ *r*) ⊕ (*snd* x' ⊗ *snd* y') ⊗ (*s* ⊗ *r'*)
using *r-distr*
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD assms*(*1*) *assms*(*2*) *eq-class-of-rng-of-frac-def*
f1 f2 mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *rel-def*
subset subset-iff)
then have (*t* ⊗ *t'*) ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (*s'* ⊗ *r* ⊕ *s* ⊗ *r'*)) =
(*t* ⊗ *t'*) ⊗ (*snd* x' ⊗ *snd* y') ⊗ (*s'* ⊗ *r*) ⊕ (*t* ⊗ *t'*) ⊗ (*snd* x' ⊗ *snd* y') ⊗ (*s* ⊗ *r'*)
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *assms*(*2*) *eq-class-of-rng-of-frac-def f1 f17 f18*
f2 m-assoc mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*)

r-distr rel-def subset subset-iff)
have *f35*:(*s* ⊗ *s'*) ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y') = (*s* ⊗ *s'*) ⊗ (*snd* y' ⊗ *fst* x') ⊕ (*s* ⊗ *s'*) ⊗ (*snd* x' ⊗ *fst* y')
using *r-distr f19 f20*
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def f1 f2*
mem-Sigma-iff partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have *f36*:(*t* ⊗ *t'*) ⊗ (*s* ⊗ *s'*) ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y') =
(*t* ⊗ *t'*) ⊗ (*s* ⊗ *s'*) ⊗ (*snd* y' ⊗ *fst* x') ⊕ (*t* ⊗ *t'*) ⊗ (*s* ⊗ *s'*) ⊗ (*snd* x' ⊗ *fst* y')
by (*smt BNF-Def.Collect-case-prodD assms*(*1*) *assms*(*2*) *eq-class-of-rng-of-frac-def f1 f17 f18 f2*
mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs(*1*) *r-distr rel-def*
subset subset-iff)
have *f37*:(*t* ⊗ *t'*) ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (*s'* ⊗ *r*) ⊖ (*s* ⊗ *s'*) ⊗ (*snd* y' ⊗ *fst* x')) ∈ *carrier R*
by (*simp add*: *f13 f14 f17 f18*)
have *f38*:(*t* ⊗ *t'*) ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (*s* ⊗ *r'*) ⊖ (*s* ⊗ *s'*) ⊗ (*snd* x' ⊗ *fst* y')) ∈ *carrier R*
using ‹*t* ⊗ *s* ⊗ *snd* x' ∈ *carrier R*› *f30 f33 f7 zero-closed*
by *auto*
have *f39*:(*t* ⊗ *t'*) ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (*s'* ⊗ *r*)) ⊖ (*t* ⊗ *t'*) ⊗ ((*s* ⊗ *s'*) ⊗ (*snd* y' ⊗ *fst* x')) ∈ *carrier R*
by (*simp add*: *f32 f37*)
have *snd* x' ⊗ *snd* y' ∈ *carrier R*

using *f1 f2 subset rev-subsetD*
by (*metis* (*no-types, lifting*) *BNF-Def.Collect-case-prodD eq-class-of-rng-of-frac-def*

mem-Sigma-iff partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*))
have $(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x'))$
$\oplus$
$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')) =$
$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r)) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ y' \otimes$
$fst\ x')) \oplus$
$(t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r')) \ominus (t \otimes t') \otimes ((s \otimes s') \otimes (snd\ x' \otimes$
$fst\ y'))$
using *f32 f33* ‹$snd\ x' \otimes snd\ y' \in carrier\ R$› ‹$t \otimes s \otimes snd\ x' \in carrier\ R$›
assms(*2*) *f17 f18 f19*
f25 f30 f5 f7 f9 l-zero r-null rel-def zero-closed
apply *clarsimp*
using *l-zero semiring-simprules*(*3*) **by** *presburger*
then have *f40*:$((t' \otimes s' \otimes snd\ y') \otimes (t \otimes (snd\ x' \otimes r \ominus s \otimes fst\ x'))) \oplus$
$((t \otimes s \otimes snd\ x') \otimes (t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y'))) =$
$((t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x'))) \oplus$
$((t \otimes t') \otimes ((snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')))$
using *f31*
by (*simp add*: *f32 f33*)
have *f41*:$(snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \in carrier$
R
by (*simp add*: *f13 f14*)
have *f42*:$(snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y') \in carrier$
R
by (*smt BNF-Def.Collect-case-prodD abelian-group.minus-closed assms*(*1*)
assms(*2*)
eq-class-of-rng-of-frac-def f1 f2 is-abelian-group mem-Sigma-iff partial-object.select-convs(*1*)

rel-def semiring-simprules(*3*) *subset subset-iff*)
then have $(t' \otimes s' \otimes snd\ y') \otimes (t \otimes (snd\ x' \otimes r \ominus s \otimes fst\ x')) \oplus$
$(t \otimes s \otimes snd\ x') \otimes (t' \otimes (snd\ y' \otimes r' \ominus s' \otimes fst\ y')) =$
$(t \otimes t') \otimes (((snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x')) \oplus$
$((snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')))$
using *r-distr*[*of* $(snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x')$
$(snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')\ t \otimes t'$]
f17 f18 f40 f41 f42
by *simp*
have $(snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \oplus (snd\ x' \otimes$
$snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y') =$
$(snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \oplus (snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes$
$(snd\ y' \otimes fst\ x') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')$
using *four-elem-comm*[*of* $(snd\ x' \otimes snd\ y') \otimes (s' \otimes r)\ (snd\ x' \otimes snd\ y') \otimes$
$(s \otimes r')\ (s \otimes s') \otimes (snd\ y' \otimes fst\ x')\ (s \otimes s') \otimes (snd\ x' \otimes fst\ y')$]
by (*smt BNF-Def.Collect-case-prodD assms eq-class-of-rng-of-frac-def f1 f2*
mem-Sigma-iff partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*)
subset subset-iff)

then have $(snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \oplus (snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y') =$
$((snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \oplus (snd\ x' \otimes snd\ y') \otimes (s \otimes r')) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')$
by *blast*
then have *f43*:$(snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \oplus (snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y') =$
$(snd\ x' \otimes snd\ y') \otimes (s' \otimes r \oplus s \otimes r') \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')$
using *f34*
by *simp*
have $(snd\ x' \otimes snd\ y') \otimes (s \otimes r') \in carrier\ R$
using ‹$snd\ x' \otimes snd\ y' \in carrier\ R$› *assms*(*2*) *f19 rel-def*
by *auto*
have $(s \otimes s') \otimes (snd\ x' \otimes fst\ y') \in carrier\ R$
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD assms eq-class-of-rng-of-frac-def f1 f2 mem-Sigma-iff partial-object.select-convs*(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have *f43bis*:$((snd\ x' \otimes snd\ y') \otimes (s' \otimes r) \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x')) \oplus ((snd\ x' \otimes snd\ y') \otimes (s \otimes r') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')) =$
$(snd\ x' \otimes snd\ y') \otimes (s' \otimes r \oplus s \otimes r') \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')$
using *a-assoc a-minus-def f41 f43*
by (*smt* ‹$snd\ x' \otimes snd\ y' \otimes (s \otimes r') \in carrier\ R$› *add.l-inv-ex add.m-closed minus-equality*)
have *f44*:$s \otimes s' \otimes (snd\ y' \otimes fst\ x') \in carrier\ R$
by (*simp add*: *f14*)
have *f45*:$s \otimes s' \otimes (snd\ x' \otimes fst\ y') \in carrier\ R$
by (*metis* (*no-types*, *lifting*) *BNF-Def.Collect-case-prodD assms eq-class-of-rng-of-frac-def f1 f2 mem-Sigma-iff partial-object.select-convs*(*1*) *rel-def semiring-simprules*(*3*) *subset subset-iff*)
then have $\ominus ((s \otimes s') \otimes (snd\ y' \otimes fst\ x') \oplus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')) =$
$\ominus ((s \otimes s') \otimes (snd\ y' \otimes fst\ x')) \ominus ((s \otimes s') \otimes (snd\ x' \otimes fst\ y'))$
using *f44 f45 inv-add*
by *auto*
then have $\ominus ((s \otimes s') \otimes (snd\ y' \otimes fst\ x') \oplus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')) =$
$\ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y')$
using *l-minus*[*of* $s \otimes s'$]
by (*simp add*: *a-minus-def f15 f16 f45*)
then have $(snd\ x' \otimes snd\ y') \otimes (s' \otimes r \oplus s \otimes r') \ominus (s \otimes s') \otimes (snd\ y' \otimes fst\ x') \ominus (s \otimes s') \otimes (snd\ x' \otimes fst\ y') =$
$(snd\ x' \otimes snd\ y') \otimes (s' \otimes r \oplus s \otimes r') \ominus ((s \otimes s') \otimes (snd\ y' \otimes fst\ x') \oplus (s \otimes s') \otimes (snd\ x' \otimes fst\ y'))$
using *right-inv-add* ‹$snd\ x' \otimes snd\ y' \in carrier\ R$› *assms*(*2*) *f13 f19 f34 f44 f45 rel-def*
by *auto*
then have $(snd\ x' \otimes snd\ y') \otimes (s' \otimes r \oplus s \otimes r') \ominus (s \otimes s') \otimes (snd\ y' \otimes fst$

x') ⊖ (s ⊗ s') ⊗ (*snd* x' ⊗ *fst* y') =
  (*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r ⊕ s ⊗ r') ⊖ ((s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y'))
  using *r-distr*
  by (*simp add*: *f35*)
 then have ((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r) ⊖ (s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x')) ⊕ ((*snd* x' ⊗ *snd* y') ⊗ (s ⊗ r') ⊖ (s ⊗ s') ⊗ (*snd* x' ⊗ *fst* y'))
  = (*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r ⊕ s ⊗ r') ⊖ ((s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y'))
  using *f43bis*
  by *simp*
 then have (t ⊗ t') ⊗ (((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r) ⊖ (s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x')) ⊕ ((*snd* x' ⊗ *snd* y') ⊗ (s ⊗ r') ⊖ (s ⊗ s') ⊗ (*snd* x' ⊗ *fst* y')))
  = (t ⊗ t') ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r ⊕ s ⊗ r') ⊖ ((s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y')))
  by *simp*
 then have (t ⊗ t') ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r) ⊖ (s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x')) ⊕
  (t ⊗ t') ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (s ⊗ r') ⊖ (s ⊗ s') ⊗ (*snd* x' ⊗ *fst* y')) =
  (t ⊗ t') ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r ⊕ s ⊗ r') ⊖ ((s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y')))
  using *r-distr*[*of* - - t ⊗ t'] *f17 f18* ‹t' ⊗ s' ⊗ *snd* y' ⊗ (t ⊗ (*snd* x' ⊗ r ⊖ s ⊗ *fst* x')) ⊕ t ⊗ s ⊗ *snd* x' ⊗ (t' ⊗ (*snd* y' ⊗ r' ⊖ s' ⊗ *fst* y')) = t ⊗ t' ⊗ (*snd* x' ⊗ *snd* y' ⊗ (s' ⊗ r) ⊖ s ⊗ s' ⊗ (*snd* y' ⊗ *fst* x') ⊕ (*snd* x' ⊗ *snd* y' ⊗ (s ⊗ r') ⊖ s ⊗ s' ⊗ (*snd* x' ⊗ *fst* y')))› *f40*
  by *auto*
 then have (t' ⊗ s' ⊗ *snd* y') ⊗ (t ⊗ (*snd* x' ⊗ r ⊖ s ⊗ *fst* x')) ⊕
  (t ⊗ s ⊗ *snd* x') ⊗ (t' ⊗ (*snd* y' ⊗ r' ⊖ s' ⊗ *fst* y')) =
  (t ⊗ t') ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r ⊕ s ⊗ r') ⊖ (s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y'))
  using *f40*
  by *simp*
 then have (t ⊗ t') ⊗ ((*snd* x' ⊗ *snd* y') ⊗ (s' ⊗ r ⊕ s ⊗ r') ⊖ (s ⊗ s') ⊗ (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y')) = **0**
  using *f5 f7*
  by (*simp add*: ‹t ⊗ s ⊗ *snd* x' ∈ *carrier R*› *f9*)
 thus *?thesis*
  using *rel-def f8*
  by *auto*
qed
then have (s' ⊗ r ⊕ s ⊗ r' $|_{rel}$ s ⊗ s') = (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y' $|_{rel}$ *snd* x' ⊗ *snd* y')
proof−
 have (s' ⊗ r ⊕ s ⊗ r', s ⊗ s') ∈ *carrier rel*
  using *assms rel-def submonoid.m-closed*
 by (*smt add.m-closed m-closed mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*)
  rev-subsetD subset)
 have (*snd* y' ⊗ *fst* x' ⊕ *snd* x' ⊗ *fst* y', *snd* x' ⊗ *snd* y') ∈ *carrier rel*

using *rel-def f1 f2 subset submonoid.m-closed eq-class-of-rng-of-frac-def*
by (*smt Product-Type.Collect-case-prodD add.m-closed mem-Sigma-iff member-class-to-carrier*
partial-object.select-convs(*1*) *semiring-simprules*(*3*) *rev-subsetD*)
thus *?thesis*
using *elem-eq-class*[*of rel*] *equiv-obj-rng-of-frac*
by (*metis* ‹$(s' \otimes r \oplus s \otimes r', s \otimes s') \mathbin{.=}_{rel} (snd\ y' \otimes fst\ x' \oplus snd\ x' \otimes fst\ y', snd\ x' \otimes snd\ y')$›
‹$(s' \otimes r \oplus s \otimes r', s \otimes s') \in carrier\ rel$› *class-of-to-rel*)
qed
thus *?thesis*
using *f3*
by *simp*
qed

lemma *closed-add-rng-of-frac*:
assumes $(r, s) \in carrier\ rel$ **and** $(r', s') \in carrier\ rel$
shows $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') \in set\text{-}class\text{-}of_{rel}$
proof−
have *f1*:$(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') = (s' \otimes r \oplus s \otimes r' \mid_{rel} s \otimes s')$
using *assms add-rng-of-frac-fundamental-lemma*
by *simp*
have *f2*:$s' \otimes r \oplus s \otimes r' \in carrier\ R$
using *assms rel-def*
by (*metis* (*no-types*, *lifting*) *add.m-closed mem-Sigma-iff monoid.m-closed monoid-axioms*
partial-object.select-convs(*1*) *rev-subsetD subset*)
have *f3*:$s \otimes s' \in S$
using *assms rel-def submonoid.m-closed*
by *simp*
from *f2* **and** *f3* **have** $(s' \otimes r \oplus s \otimes r', s \otimes s') \in carrier\ rel$
by (*simp add*: *rel-def*)
thus *?thesis*
using *set-eq-class-of-rng-of-frac-def f1*
by *auto*
qed

lemma *closed-rel-add*:
assumes $(r, s) \in carrier\ rel$ **and** $(r', s') \in carrier\ rel$
shows $(s' \otimes r \oplus s \otimes r', s \otimes s') \in carrier\ rel$
proof−
have $s \otimes s' \in S$
using *assms rel-def submonoid.m-closed*
by *simp*
have $s' \otimes r \oplus s \otimes r' \in carrier\ R$
using *assms rel-def*
by (*metis* (*no-types*, *lifting*) *add.m-closed mem-Sigma-iff monoid.m-closed monoid-axioms*
partial-object.select-convs(*1*) *rev-subsetD subset*)

thus *?thesis*
using *rel-def*
by (*simp add*: ‹$s \otimes s' \in S$›)
qed

lemma *assoc-add-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel* **and** $(r', s') \in$ *carrier rel* **and** $(r'', s'') \in$ *carrier rel*
shows $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') =$
$(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} ((r' \mid_{rel} s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s''))$
proof−
have $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') = (s' \otimes r \oplus s \otimes r' \mid_{rel} s \otimes s')$
using *assms*(*1*) *assms*(*2*) *add-rng-of-frac-fundamental-lemma*
by *simp*
then have *f1*:$(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') =$
$(s'' \otimes (s' \otimes r \oplus s \otimes r') \oplus (s \otimes s') \otimes r'' \mid_{rel} (s \otimes s') \otimes s'')$
using *assms add-rng-of-frac-fundamental-lemma closed-rel-add*
by *simp*
have $(r' \mid_{rel} s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') = (s'' \otimes r' \oplus s' \otimes r'' \mid_{rel} s' \otimes s'')$
using *assms*(*2*) *assms*(*3*) *add-rng-of-frac-fundamental-lemma*
by *simp*
then have *f2*:$(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} ((r' \mid_{rel} s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s''))$
$=$
$((s' \otimes s'') \otimes r \oplus s \otimes (s'' \otimes r' \oplus s' \otimes r'') \mid_{rel} s \otimes (s' \otimes s''))$
using *assms add-rng-of-frac-fundamental-lemma closed-rel-add*
by *simp*
have *f3*:$(s \otimes s') \otimes s'' = s \otimes (s' \otimes s'')$
using *m-assoc subset assms rel-def*
by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*) *rev-subsetD*)
have $s'' \otimes (s' \otimes r \oplus s \otimes r') \oplus (s \otimes s') \otimes r'' = (s' \otimes s'') \otimes r \oplus s \otimes (s'' \otimes r'$
$\oplus s' \otimes r'')$
by (*smt a-assoc assms m-comm mem-Sigma-iff monoid.m-assoc monoid.m-closed*
monoid-axioms
partial-object.select-convs(*1*) *r-distr rel-def subset subset-iff*)
thus *?thesis*
using *f1 f2 f3*
by *simp*
qed

lemma *add-rng-of-frac-zero*:
shows $(\mathbf{0} \mid_{rel} \mathbf{1}) \in$ *set-class-of*$_{rel}$
by (*metis* (*no-types*, *lifting*) *closed-mult-rng-of-frac mem-Sigma-iff monoid.simps*(*2*)
one-closed
partial-object.select-convs(*1*) *rec-monoid-rng-of-frac-def rel-def right-unit-mult-rng-of-frac*
semiring-simprules(*4*) *zero-closed*)

lemma *l-unit-add-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel*
shows $\mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac} \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r \mid_{rel} s) = (r \mid_{rel} s)$
proof−

have $(\mathbf{0}\ |_{rel}\ \mathbf{1}) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) = (s \otimes \mathbf{0} \oplus \mathbf{1} \otimes r\ |_{rel}\ \mathbf{1} \otimes s)$
using *assms add-rng-of-frac-fundamental-lemma*
by (*simp add*: *rel-def*)
then have $(\mathbf{0}\ |_{rel}\ \mathbf{1}) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) = (r\ |_{rel}\ s)$
using *assms rel-def subset*
by *auto*
thus *?thesis*
using *rec-rng-of-frac-def*
by *simp*
qed

lemma *r-unit-add-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel*
shows $(r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac} = (r\ |_{rel}\ s)$
proof−
have $(r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (\mathbf{0}\ |_{rel}\ \mathbf{1}) = (\mathbf{1} \otimes r \oplus s \otimes \mathbf{0}\ |_{rel}\ s \otimes \mathbf{1})$
using *assms add-rng-of-frac-fundamental-lemma*
by (*simp add*: *rel-def*)
then have $(r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (\mathbf{0}\ |_{rel}\ \mathbf{1}) = (r\ |_{rel}\ s)$
using *assms rel-def subset*
by *auto*
thus *?thesis*
using *rec-rng-of-frac-def*
by *simp*
qed

lemma *comm-add-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel* **and** $(r', s') \in$ *carrier rel*
shows $(r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (r'\ |_{rel}\ s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s)$
proof−
have *f1*:$(r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (s' \otimes r \oplus s \otimes r'\ |_{rel}\ s \otimes s')$
using *assms add-rng-of-frac-fundamental-lemma*
by *simp*
have *f2*:$(r'\ |_{rel}\ s') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) = (s \otimes r' \oplus s' \otimes r\ |_{rel}\ s' \otimes s)$
using *assms add-rng-of-frac-fundamental-lemma*
by *simp*
thus *?thesis*
using *f1 f2*
by (*metis* (*no-types*, *lifting*) *add.m-comm assms*(*1*) *assms*(*2*) *m-comm mem-Sigma-iff*

partial-object.select-convs(*1*) *rel-def semiring-simprules*(*3*) *rev-subsetD sub-set*)
qed

lemma *class-of-zero-rng-of-frac*:
assumes $s \in S$
shows $(\mathbf{0}\ |_{rel}\ s) = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$
proof−
have *f1*:$(\mathbf{0}, s) \in$ *carrier rel*

using *assms rel-def*
by *simp*
have $\mathbf{1} \otimes (\mathbf{1} \otimes \mathbf{0} \ominus s \otimes \mathbf{0}) = \mathbf{0}$
using *assms local.ring-axioms rev-subsetD ring.ring-simprules(14) subset*
by *fastforce*
then have $(\mathbf{0}, s) \;.{=}_{rel}\; (\mathbf{0}, \mathbf{1})$
using *rel-def submonoid.one-closed*
by *auto*
thus *?thesis*
using *elem-eq-class equiv-obj-rng-of-frac f1 rec-rng-of-frac-def*
by (*metis* (*no-types*, *lifting*) *class-of-to-rel mem-Sigma-iff one-closed partial-object.select-convs*(*1*)

rel-def ring-record-simps(*11*))
qed

lemma *r-inv-add-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel*
shows $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (\ominus r \mid_{rel} s) = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$
proof−
have $(\ominus r, s) \in$ *carrier rel*
using *assms rel-def*
by *simp*
then have $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (\ominus r \mid_{rel} s) = (s \otimes r \oplus s \otimes \ominus r \mid_{rel} s \otimes s)$
using *assms add-rng-of-frac-fundamental-lemma*
by *simp*
then have $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (\ominus r \mid_{rel} s) = (\mathbf{0} \mid_{rel} s \otimes s)$
using *r-minus*[*of s r*] *assms rel-def subset rev-subsetD r-neg*
by *auto*
thus *?thesis*
using *class-of-zero-rng-of-frac assms rel-def submonoid.m-closed*
by *simp*
qed

lemma *l-inv-add-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel*
shows $(\ominus r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r \mid_{rel} s) = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$
proof−
have $(\ominus r, s) \in$ *carrier rel*
using *assms rel-def*
by *simp*
then have $(\ominus r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r \mid_{rel} s) = (s \otimes \ominus r \oplus s \otimes r \mid_{rel} s \otimes s)$
using *assms add-rng-of-frac-fundamental-lemma*
by *simp*
then have $(\ominus r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r \mid_{rel} s) = (\mathbf{0} \mid_{rel} s \otimes s)$
using *r-minus*[*of s r*] *assms rel-def subset rev-subsetD l-neg*
by *auto*
thus *?thesis*
using *class-of-zero-rng-of-frac assms rel-def submonoid.m-closed*
by *simp*

qed

lemma *abelian-group-rng-of-frac*:
shows *abelian-group* (*rec-rng-of-frac*)
proof
show $\bigwedge x\ y.\ [\![x \in$ *carrier* (*add-monoid rec-rng-of-frac*);
$y \in$ *carrier* (*add-monoid rec-rng-of-frac*)$]\!]$
$\Longrightarrow x \otimes_{\textit{add-monoid rec-rng-of-frac}} y$
$\in$ *carrier* (*add-monoid rec-rng-of-frac*)
using *closed-add-rng-of-frac*
by (*smt mem-Collect-eq monoid.select-convs*(*1*) *partial-object.select-convs*(*1*) *rec-rng-of-frac-def*
set-eq-class-of-rng-of-frac-def)
show $\bigwedge x\ y\ z.$
$[\![x \in$ *carrier* (*add-monoid rec-rng-of-frac*);
$y \in$ *carrier* (*add-monoid rec-rng-of-frac*);
$z \in$ *carrier* (*add-monoid rec-rng-of-frac*)$]\!]$
$\Longrightarrow x \otimes_{\textit{add-monoid rec-rng-of-frac}} y \otimes_{\textit{add-monoid rec-rng-of-frac}} z =$
$x \otimes_{\textit{add-monoid rec-rng-of-frac}} (y \otimes_{\textit{add-monoid rec-rng-of-frac}} z)$
using *assoc-add-rng-of-frac*
by (*smt mem-Collect-eq monoid.simps*(*1*) *partial-object.select-convs*(*1*) *rec-rng-of-frac-def*

set-eq-class-of-rng-of-frac-def)
show $\mathbf{1}_{\textit{add-monoid rec-rng-of-frac}} \in$ *carrier* (*add-monoid rec-rng-of-frac*)
using *add-rng-of-frac-zero* **by** (*simp add*: *rec-rng-of-frac-def*)
show $\bigwedge x.\ x \in$ *carrier* (*add-monoid rec-rng-of-frac*) $\Longrightarrow$
$\mathbf{1}_{\textit{add-monoid rec-rng-of-frac}} \otimes_{\textit{add-monoid rec-rng-of-frac}} x = x$
using *l-unit-add-rng-of-frac*
by (*smt mem-Collect-eq monoid.select-convs*(*1*) *monoid.select-convs*(*2*) *partial-object.select-convs*(*1*)
rec-rng-of-frac-def set-eq-class-of-rng-of-frac-def)
show $\bigwedge x.\ x \in$ *carrier* (*add-monoid rec-rng-of-frac*) $\Longrightarrow$
$x \otimes_{\textit{add-monoid rec-rng-of-frac}} \mathbf{1}_{\textit{add-monoid rec-rng-of-frac}} = x$
using *r-unit-add-rng-of-frac*
by (*smt mem-Collect-eq monoid.select-convs*(*1*) *monoid.select-convs*(*2*) *partial-object.select-convs*(*1*)
rec-rng-of-frac-def set-eq-class-of-rng-of-frac-def)
show $\bigwedge x\ y.\ [\![x \in$ *carrier* (*add-monoid rec-rng-of-frac*); $y \in$ *carrier* (*add-monoid rec-rng-of-frac*)$]\!]$
$\Longrightarrow x \otimes_{\textit{add-monoid rec-rng-of-frac}} y = y \otimes_{\textit{add-monoid rec-rng-of-frac}} x$
using *comm-add-rng-of-frac*
by (*smt mem-Collect-eq monoid.select-convs*(*1*) *partial-object.select-convs*(*1*) *rec-rng-of-frac-def*
set-eq-class-of-rng-of-frac-def)
show *carrier* (*add-monoid rec-rng-of-frac*) $\subseteq$ *Units* (*add-monoid rec-rng-of-frac*)
proof
show $x \in$ *Units* (*add-monoid rec-rng-of-frac*) **if** $x \in$ *carrier* (*add-monoid rec-rng-of-frac*) **for** x
proof−

have $x \in$ *set-class-of*$_{rel}$
using *that rec-rng-of-frac-def* **by** *simp*
then obtain r **and** s **where** *f1*:$(r, s) \in$ *carrier rel* **and** *f2*:$x = (r \mid_{rel} s)$
using *set-eq-class-of-rng-of-frac-def*
by (*smt mem-Collect-eq*)
then have *f3*:$(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (\ominus\ r \mid_{rel} s) = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$
using *f1 r-inv-add-rng-of-frac*[*of r s*]
by *simp*
have *f4*:$(\ominus\ r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r \mid_{rel} s) = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$
using *f1 l-inv-add-rng-of-frac*[*of r s*]
by *simp*
then have $\exists\, y \in$ *set-class-of*$_{rel}$. $y \oplus_{rec\text{-}rng\text{-}of\text{-}frac} x = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac} \wedge x \oplus_{rec\text{-}rng\text{-}of\text{-}frac} y = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$
using *f2 f3 f4*
by (*metis* (*no-types*, *lifting*) *abelian-group.a-inv-closed class-of-zero-rng-of-frac*

closed-add-rng-of-frac f1 is-abelian-group mem-Sigma-iff partial-object.select-convs(*1*)

rel-def r-unit-add-rng-of-frac zero-closed)
thus $x \in$ *Units* (*add-monoid rec-rng-of-frac*)
using *rec-rng-of-frac-def that* **by** (*simp add*: *Units-def*)
qed
qed
qed

lemma *r-distr-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel* **and** $(r', s') \in$ *carrier rel* **and** $(r'', s'') \in$ *carrier rel*
shows $((r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s')) \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') =$
$(r \mid_{rel} s) \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'')$
proof−
have $(r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') = (s' \otimes r \oplus s \otimes r' \mid_{rel} s \otimes s')$
using *assms*(*1*) *assms*(*2*) *add-rng-of-frac-fundamental-lemma*
by *simp*
then have *f1*:$((r \mid_{rel} s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s')) \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') =$
$((s' \otimes r \oplus s \otimes r') \otimes r'' \mid_{rel} (s \otimes s') \otimes s'')$
using *assms mult-rng-of-frac-fundamental-lemma*
by (*simp add*: *closed-rel-add rec-monoid-rng-of-frac-def rec-rng-of-frac-def*)
have *f2*:$(r \mid_{rel} s) \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') = (r \otimes r'' \mid_{rel} s \otimes s'')$
using *assms*(*1*) *assms*(*3*) *mult-rng-of-frac-fundamental-lemma*
by (*simp add*: *rec-monoid-rng-of-frac-def rec-rng-of-frac-def*)
have *f3*:$(r' \mid_{rel} s') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') = (r' \otimes r'' \mid_{rel} s' \otimes s'')$
using *assms*(*2*) *assms*(*3*) *mult-rng-of-frac-fundamental-lemma*
by (*simp add*: *rec-monoid-rng-of-frac-def rec-rng-of-frac-def*)
have *f4*:$(r \otimes r'', s \otimes s'') \in$ *carrier rel*
using *rel-def assms*(*1*) *assms*(*3*) *submonoid.m-closed*
by *simp*
have *f5*:$(r' \otimes r'', s' \otimes s'') \in$ *carrier rel*

using *rel-def assms*(*2*) *assms*(*3*) *submonoid.m-closed*
by *simp*
from *f2* **and** *f3* **have** *f6*:$(r \mid_{rel} s) \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'') \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r' \mid_{rel} s') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'' \mid_{rel} s'')$
$= ((s' \otimes s'') \otimes (r \otimes r'') \oplus (s \otimes s'') \otimes (r' \otimes r'') \mid_{rel} (s \otimes s'') \otimes (s' \otimes s''))$
using *assms f4 f5 submonoid.m-closed add-rng-of-frac-fundamental-lemma*
by *simp*
have $(s \otimes s'' \otimes (s' \otimes s'')) \otimes ((s' \otimes r \oplus s \otimes r') \otimes r'') = (s \otimes s'' \otimes (s' \otimes s'')) \otimes (s' \otimes r \ \otimes r'' \oplus s \otimes r' \otimes r'')$
using *assms rel-def subset rev-subsetD l-distr*
by (*smt mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*))
then have *f7*:$(s \otimes s'' \otimes (s' \otimes s'')) \otimes ((s' \otimes r \oplus s \otimes r') \otimes r'') =$
$(s \otimes s'' \otimes (s' \otimes s'')) \otimes (s' \otimes r \ \otimes r'') \oplus (s \otimes s'' \otimes (s' \otimes s'')) \otimes (s \otimes r' \otimes r'')$
using *assms rel-def subset rev-subsetD submonoid.m-closed r-distr*
by (*smt mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*))
have *f8*:$(s \otimes s' \otimes s'') \otimes (s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r'')) =$
$(s \otimes s' \otimes s'') \otimes (s' \otimes s'' \otimes (r \otimes r'')) \oplus (s \otimes s' \otimes s'') \otimes (s \otimes s'' \otimes (r' \otimes r''))$
using *assms rel-def subset rev-subsetD submonoid.m-closed r-distr*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
have $(s \otimes s'' \otimes (s' \otimes s'')) = (s \otimes (s'' \otimes s') \otimes s'')$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-assoc*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
then have *f9*:$(s \otimes s'' \otimes (s' \otimes s'')) = (s \otimes s' \otimes (s'' \otimes s''))$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-comm m-assoc*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
then have *f10*:$(s \otimes s'' \otimes (s' \otimes s'')) \otimes (s' \otimes r \ \otimes r'') = (s \otimes s' \otimes s'') \otimes (s' \otimes s'' \otimes (r \otimes r''))$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-assoc m-comm*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
have $(s \otimes s'' \otimes (r' \otimes r'')) = (s'' \otimes s \otimes (r' \otimes r''))$
using *assms rel-def subset rev-subsetD m-comm*
by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*))
then have $(s \otimes s'' \otimes (s' \otimes s'')) \otimes (s \otimes r' \otimes r'') = (s \otimes s' \otimes s'') \otimes (s \otimes s'' \otimes (r' \otimes r''))$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-comm m-assoc f9*
by (*smt mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*))
then have $((s \otimes s'' \otimes (s' \otimes s'')) \otimes ((s' \otimes r \oplus s \otimes r') \otimes r'') = (s \otimes s' \otimes s'') \otimes (s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r'')))$
using *f7 f8 f10*
by *presburger*
then have $((s \otimes s'' \otimes (s' \otimes s'')) \otimes ((s' \otimes r \oplus s \otimes r') \otimes r'') \ominus (s \otimes s' \otimes s'') \otimes (s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r''))) = \mathbf{0}$
by (*smt a-minus-def assms*(*1*) *assms*(*2*) *assms*(*3*) *closed-rel-add mem-Sigma-iff*

partial-object.select-convs(*1*) *r-neg rel-def semiring-simprules*(*3*) *rev-subsetD subset*)
then have *f11*:$\mathbf{1} \otimes (((s \otimes s'' \otimes (s' \otimes s'')) \otimes ((s' \otimes r \oplus s \otimes r') \otimes r'') \ominus (s \otimes s' \otimes s'') \otimes (s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r'')))) = \mathbf{0}$
by *simp*

have *f12*:$((s' \otimes r \oplus s \otimes r') \otimes r'', s \otimes s' \otimes s'') \in$ *carrier rel*
using *assms closed-rel-add rel-def*
by *auto*
have *f13*:$(s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r''), s \otimes s'' \otimes (s' \otimes s'')) \in$ *carrier rel*
by (*simp add*: *closed-rel-add f4 f5*)
have $\mathbf{1} \in S$
using *submonoid.one-closed*
by *simp*
then have $((s' \otimes r \oplus s \otimes r') \otimes r'', s \otimes s' \otimes s'') .=_{rel} (s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r''), s \otimes s'' \otimes (s' \otimes s''))$
using *rel-def f11 f13 f12*
by *auto*
then have $((s' \otimes r \oplus s \otimes r') \otimes r''|_{rel}\ s \otimes s' \otimes s'') = (s' \otimes s'' \otimes (r \otimes r'') \oplus s \otimes s'' \otimes (r' \otimes r'')\ |_{rel}\ s \otimes s'' \otimes (s' \otimes s''))$
using *elem-eq-class*
by (*metis class-of-to-rel equiv-obj-rng-of-frac f12 f13*)
thus *?thesis*
using *f1 f6*
by *simp*
qed

lemma *l-distr-rng-of-frac*:
assumes $(r, s) \in$ *carrier rel* **and** $(r', s') \in$ *carrier rel* **and** $(r'', s'') \in$ *carrier rel*
shows $(r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} ((r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s')) =$
$(r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s')$
proof−
have $(r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (s' \otimes r \oplus s \otimes r'\ |_{rel}\ s \otimes s')$
using *assms*(*1*) *assms*(*2*) *add-rng-of-frac-fundamental-lemma*
by *simp*
then have *f1*:$(r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} ((r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s')) =$
$(r'' \otimes (s' \otimes r \oplus s \otimes r')\ |_{rel}\ s'' \otimes (s \otimes s'))$
using *assms mult-rng-of-frac-fundamental-lemma*
by (*simp add*: *closed-rel-add rec-monoid-rng-of-frac-def rec-rng-of-frac-def*)
have *f2*:$(r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) = (r'' \otimes r\ |_{rel}\ s'' \otimes s)$
using *assms*(*1*) *assms*(*3*) *mult-rng-of-frac-fundamental-lemma*
by (*simp add*: *rec-monoid-rng-of-frac-def rec-rng-of-frac-def*)
have *f3*:$(r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r'\ |_{rel}\ s') = (r'' \otimes r'\ |_{rel}\ s'' \otimes s')$
using *assms*(*2*) *assms*(*3*) *mult-rng-of-frac-fundamental-lemma*
by (*simp add*: *rec-monoid-rng-of-frac-def rec-rng-of-frac-def*)
have *f4*:$(r'' \otimes r, s'' \otimes s) \in$ *carrier rel*
using *rel-def assms*(*1*) *assms*(*3*) *submonoid.m-closed*
by *simp*
have *f5*:$(r'' \otimes r', s'' \otimes s') \in$ *carrier rel*
using *rel-def assms*(*2*) *assms*(*3*) *submonoid.m-closed*
by *simp*
from *f2* **and** *f3* **have** *f6*:$(r''\ |_{rel}\ s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r\ |_{rel}\ s) \oplus_{rec\text{-}rng\text{-}of\text{-}frac}$

$(r'' |_{rel} s'') \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (r' |_{rel} s')$
$= ((s'' \otimes s') \otimes (r'' \otimes r) \oplus (s'' \otimes s) \otimes (r'' \otimes r') |_{rel} (s'' \otimes s) \otimes (s'' \otimes s'))$
using *assms f4 f5 submonoid.m-closed add-rng-of-frac-fundamental-lemma*
by *simp*
have $(s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r \oplus s \otimes r')) = (s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r) \oplus r'' \otimes (s \otimes r'))$
using *assms rel-def subset rev-subsetD r-distr*
by (*smt mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*))
then have *f7*:$(s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r \oplus s \otimes r')) =$
$(s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r)) \oplus (s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s \otimes r'))$
using *assms rel-def subset rev-subsetD submonoid.m-closed r-distr*
by (*smt mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*))
have *f8*:$(s'' \otimes s \otimes s') \otimes (s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r')) =$
$(s'' \otimes s \otimes s') \otimes (s'' \otimes s' \otimes (r'' \otimes r)) \oplus (s'' \otimes s \otimes s') \otimes (s'' \otimes s \otimes (r'' \otimes r'))$
using *assms rel-def subset rev-subsetD submonoid.m-closed r-distr*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
have $(s'' \otimes s \otimes (s'' \otimes s')) = (s'' \otimes (s \otimes s'') \otimes s')$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-assoc*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
then have *f9*:$(s'' \otimes s \otimes (s'' \otimes s')) = (s'' \otimes s'' \otimes (s \otimes s'))$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-comm m-assoc*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
then have *f10*:$(s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes s' \otimes r) = (s'' \otimes s \otimes s') \otimes (s'' \otimes s' \otimes (r'' \otimes r))$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-assoc m-comm*
by (*smt mem-Sigma-iff partial-object.select-convs*(*1*) *semiring-simprules*(*3*))
have $(s'' \otimes s \otimes (r'' \otimes r')) = (s \otimes s'' \otimes (r'' \otimes r'))$
using *assms rel-def subset rev-subsetD m-comm*
by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*))
then have $(s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes s \otimes r') = (s'' \otimes s \otimes s') \otimes (s'' \otimes s \otimes (r'' \otimes r'))$
using *assms rel-def subset rev-subsetD submonoid.m-closed m-comm m-assoc f9*
by (*smt mem-Sigma-iff monoid.m-closed monoid-axioms partial-object.select-convs*(*1*))
then have $((s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r \oplus s \otimes r')) = (s'' \otimes (s \otimes s')) \otimes (s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r')))$
using *f7 f8 f10*
by (*smt assms*(*1*) *assms*(*2*) *assms*(*3*) *m-assoc mem-Sigma-iff partial-object.select-convs*(*1*) *rel-def*
rev-subsetD subset)
then have $((s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r \oplus s \otimes r')) \ominus (s'' \otimes (s \otimes s')) \otimes (s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r'))) = \mathbf{0}$
by (*smt a-minus-def assms*(*1*) *assms*(*2*) *assms*(*3*) *closed-rel-add mem-Sigma-iff partial-object.select-convs*(*1*)
r-neg rel-def semiring-simprules(*3*) *rev-subsetD subset*)
then have *f11*:$\mathbf{1} \otimes (((s'' \otimes s \otimes (s'' \otimes s')) \otimes (r'' \otimes (s' \otimes r \oplus s \otimes r')) \ominus (s'' \otimes (s \otimes s')) \otimes (s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r')))) = \mathbf{0}$
by *simp*
have *f12*:$(r'' \otimes (s' \otimes r \oplus s \otimes r'), s'' \otimes (s \otimes s')) \in$ *carrier rel*

using *assms closed-rel-add rel-def*
by *auto*
have *f13*:$(s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r'), s'' \otimes s \otimes (s'' \otimes s')) \in$ *carrier rel*
by (*simp add*: *closed-rel-add f4 f5*)
have $\mathbf{1} \in S$
using *submonoid.one-closed*
by *simp*
then have $(r'' \otimes (s' \otimes r \oplus s \otimes r'), s'' \otimes (s \otimes s')) .=_{rel} (s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r'), s'' \otimes s \otimes (s'' \otimes s'))$
using *rel-def f11 f13 f12*
by *auto*
then have $(r'' \otimes (s' \otimes r \oplus s \otimes r') |_{rel}\ s'' \otimes (s \otimes s')) = (s'' \otimes s' \otimes (r'' \otimes r) \oplus s'' \otimes s \otimes (r'' \otimes r') |_{rel}\ s'' \otimes s \otimes (s'' \otimes s'))$
using *elem-eq-class*
by (*metis class-of-to-rel equiv-obj-rng-of-frac f12 f13*)
thus *?thesis*
using *f1 f6*
by *simp*
qed

lemma *rng-rng-of-frac*:
shows *ring* (*rec-rng-of-frac*)
proof−
have *f1*:$\forall x\ y\ z.\ x \in$ *carrier rec-rng-of-frac* $\longrightarrow y \in$ *carrier rec-rng-of-frac* $\longrightarrow z \in$ *carrier rec-rng-of-frac*
$\longrightarrow (x \oplus_{rec\text{-}rng\text{-}of\text{-}frac} y) \otimes_{rec\text{-}rng\text{-}of\text{-}frac} z = x \otimes_{rec\text{-}rng\text{-}of\text{-}frac} z \oplus_{rec\text{-}rng\text{-}of\text{-}frac} y \otimes_{rec\text{-}rng\text{-}of\text{-}frac} z$
using *r-distr-rng-of-frac rec-rng-of-frac-def*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *set-eq-class-of-rng-of-frac-def*)
have *f2*:$\forall x\ y\ z.\ x \in$ *carrier rec-rng-of-frac* $\longrightarrow y \in$ *carrier rec-rng-of-frac* $\longrightarrow z \in$ *carrier rec-rng-of-frac*
$\longrightarrow z \otimes_{rec\text{-}rng\text{-}of\text{-}frac} (x \oplus_{rec\text{-}rng\text{-}of\text{-}frac} y) = z \otimes_{rec\text{-}rng\text{-}of\text{-}frac} x \oplus_{rec\text{-}rng\text{-}of\text{-}frac} z \otimes_{rec\text{-}rng\text{-}of\text{-}frac} y$
using *l-distr-rng-of-frac rec-rng-of-frac-def*
by (*smt mem-Collect-eq partial-object.select-convs*(*1*) *set-eq-class-of-rng-of-frac-def*)
then have *ring-axioms* (*rec-rng-of-frac*)
using *ring-axioms-def f1 f2*
by *auto*
thus *?thesis*
using *ring-def*[*of rec-rng-of-frac*] *abelian-group-rng-of-frac monoid-rng-of-frac rec-rng-of-frac-def*
abelian-group-axioms-def rec-monoid-rng-of-frac-def eq-class-of-rng-of-frac-def
by (*simp add*: *Group.monoid-def*)
qed

lemma *crng-rng-of-frac*:
shows *cring* (*rec-rng-of-frac*)
using *cring-def*[*of rec-rng-of-frac*] *rng-rng-of-frac comm-monoid-rng-of-frac rec-rng-of-frac-def*

rec-monoid-rng-of-frac-def eq-class-of-rng-of-frac-def
by (*metis* (*no-types*, *lifting*) *comm-monoid.m-comm monoid.monoid-comm-monoidI monoid.select-convs*(*1*)
partial-object.select-convs(*1*) *ring.is-monoid*)

lemma *simp-in-frac*:
assumes $(r,\ s) \in$ *carrier rel* **and** $s' \in S$
shows $(r\ |_{rel}\ s) = (s' \otimes r\ |_{rel}\ s' \otimes s)$
proof −
have *f1*:$(s' \otimes r,\ s' \otimes s) \in$ *carrier rel*
using *assms rel-def submonoid.m-closed subset rev-subsetD*
by *auto*
have $(s' \otimes s) \otimes r \ominus s \otimes (s' \otimes r) = (s' \otimes s) \otimes r \ominus (s \otimes s') \otimes r$
using *assms subset rev-subsetD m-assoc*[*of s s' r*] *rel-def*
by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*))
then have $(s' \otimes s) \otimes r \ominus s \otimes (s' \otimes r) = (s' \otimes s) \otimes r \ominus (s' \otimes s) \otimes r$
using *m-comm*[*of s s'*] *assms subset rev-subsetD rel-def*
by (*metis* (*no-types*, *lifting*) *mem-Sigma-iff partial-object.select-convs*(*1*))
then have $(s' \otimes s) \otimes r \ominus s \otimes (s' \otimes r) = \mathbf{0}$
by (*metis* (*no-types*, *lifting*) *a-minus-def assms mem-Sigma-iff partial-object.select-convs*(*1*)

r-neg rel-def semiring-simprules(*3*) *rev-subsetD subset*)
then have $\mathbf{1} \otimes ((s' \otimes s) \otimes r \ominus s \otimes (s' \otimes r)) = \mathbf{0}$
by *simp*
then have $(r,\ s)\ .=_{rel}\ (s' \otimes r,\ s' \otimes s)$
using *assms*(*1*) *f1 rel-def one-closed*
by *auto*
thus *?thesis*
using *elem-eq-class*
by (*metis assms*(*1*) *class-of-to-rel equiv-obj-rng-of-frac f1*)
qed

1.2 The Natural Homomorphism from a Ring to Its Localization

definition *rng-to-rng-of-frac* :: $'a \Rightarrow ('a \times 'a)$ *set* **where**
rng-to-rng-of-frac $r \equiv (r\ |_{rel}\ \mathbf{1})$

lemma *rng-to-rng-of-frac-is-ring-hom* :
shows *rng-to-rng-of-frac* $\in$ *ring-hom R rec-rng-of-frac*
proof −
have *f1*:*rng-to-rng-of-frac* $\in$ *carrier R* $\rightarrow$ *carrier rec-rng-of-frac*
using *rng-to-rng-of-frac-def rec-rng-of-frac-def set-eq-class-of-rng-of-frac-def rel-def*
by *fastforce*
have *f2*:$\forall x\ y.\ x \in$ *carrier R* $\wedge\ y \in$ *carrier R*
$\longrightarrow$ *rng-to-rng-of-frac* $(x \otimes_R y) =$ *rng-to-rng-of-frac* $x \otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac* y

proof(*rule allI*, *rule allI*, *rule impI*)
fix *x y*
assume *x* ∈ *carrier R* ∧ *y* ∈ *carrier R*
have *f1*:*rng-to-rng-of-frac* (*x* $\otimes_R$ *y*) = (*x* ⊗ *y* $|_{rel}$ **1**)
using *rng-to-rng-of-frac-def*
by *simp*
have *rng-to-rng-of-frac x* $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y* = (*x* $|_{rel}$ **1**) $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ (*y* $|_{rel}$ **1**)
using *rng-to-rng-of-frac-def*
by *simp*
then have *rng-to-rng-of-frac x* $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y* = (*x* ⊗ *y* $|_{rel}$ **1**)
using *mult-rng-of-frac-fundamental-lemma*
by (*simp add*: ‹*x* ∈ *carrier R* ∧ *y* ∈ *carrier R*› *rec-monoid-rng-of-frac-def rec-rng-of-frac-def rel-def*)
thus *rng-to-rng-of-frac* (*x* $\otimes_R$ *y*) = *rng-to-rng-of-frac x* $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y*
using *f1*
by *auto*
qed
have *f3*:∀ *x y*. *x* ∈ *carrier R* ∧ *y* ∈ *carrier R*
⟶ *rng-to-rng-of-frac* (*x* $\oplus_R$ *y*) = *rng-to-rng-of-frac x* $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y*
proof(*rule allI*, *rule allI*, *rule impI*)
fix *x y*
assume *a*:*x* ∈ *carrier R* ∧ *y* ∈ *carrier R*
have *f1*:*rng-to-rng-of-frac* (*x* $\oplus_R$ *y*) = (*x* ⊕ *y* $|_{rel}$ **1**)
using *rng-to-rng-of-frac-def*
by *simp*
have *rng-to-rng-of-frac x* $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y* = (*x* $|_{rel}$ **1**) $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ (*y* $|_{rel}$ **1**)
using *rng-to-rng-of-frac-def*
by *simp*
then have *rng-to-rng-of-frac x* $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y* = (**1** ⊗ *x* ⊕ **1** ⊗ *y* $|_{rel}$ **1** ⊗ **1**)
using *mult-rng-of-frac-fundamental-lemma a*
eq-obj-rng-of-frac.add-rng-of-frac-fundamental-lemma eq-obj-rng-of-frac.rng-to-rng-of-frac-def
eq-obj-rng-of-frac-axioms f1
by *fastforce*
then have *rng-to-rng-of-frac x* $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y* = (*x* ⊕ *y* $|_{rel}$ **1**)
using *l-one a*
by *simp*
thus *rng-to-rng-of-frac* (*x* $\oplus_R$ *y*) = *rng-to-rng-of-frac x* $\oplus_{rec\text{-}rng\text{-}of\text{-}frac}$ *rng-to-rng-of-frac y*
using *f1*
by *auto*
qed

have *rng-to-rng-of-frac* $\mathbf{1}$ = ($\mathbf{1}$ $|_{rel}$ $\mathbf{1}$)
  using *rng-to-rng-of-frac-def*
  by *simp*
then have *f4*:*rng-to-rng-of-frac* $\mathbf{1}_R$ = $\mathbf{1}_{rec\text{-}rng\text{-}of\text{-}frac}$
  using *rec-rng-of-frac-def*
  by *simp*
thus *?thesis*
  using *ring-hom-def*[*of R rec-rng-of-frac*] *f1 f2 f3 f4*
  by *simp*
qed

lemma *Im-rng-to-rng-of-frac-unit*:
assumes *x* ∈ *rng-to-rng-of-frac* ‘ *S*
shows *x* ∈ *Units rec-rng-of-frac*
proof−
obtain *s* **where** *a1*:*s* ∈ *S* **and** *a2*:*x* = (*s* $|_{rel}$ $\mathbf{1}$)
  using *assms rng-to-rng-of-frac-def rel-def*
  by *auto*
then have (*s* $|_{rel}$ $\mathbf{1}$) $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ ($\mathbf{1}$ $|_{rel}$ *s*) = (*s* ⊗ $\mathbf{1}$ $|_{rel}$ *s* ⊗ $\mathbf{1}$)
  using *mult-rng-of-frac-fundamental-lemma rec-monoid-rng-of-frac-def rec-rng-of-frac-def rel-def subset*
  by *auto*
then have *f1*:(*s* $|_{rel}$ $\mathbf{1}$) $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ ($\mathbf{1}$ $|_{rel}$ *s*) = ($\mathbf{1}$ $|_{rel}$ $\mathbf{1}$)
  using *simp-in-frac a1 rel-def*
  by *auto*
have ($\mathbf{1}$ $|_{rel}$ *s*) $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ (*s* $|_{rel}$ $\mathbf{1}$) = (*s* ⊗ $\mathbf{1}$ $|_{rel}$ *s* ⊗ $\mathbf{1}$)
  using *mult-rng-of-frac-fundamental-lemma rec-monoid-rng-of-frac-def rec-rng-of-frac-def rel-def*
    subset a1
  by *auto*
then have *f2*:($\mathbf{1}$ $|_{rel}$ *s*) $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ (*s* $|_{rel}$ $\mathbf{1}$) = ($\mathbf{1}$ $|_{rel}$ $\mathbf{1}$)
  using *simp-in-frac a1 rel-def*
  by *auto*
then have *f3*:∃ *y*∈*carrier rec-rng-of-frac*. *y* $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ *x* = $\mathbf{1}_{rec\text{-}rng\text{-}of\text{-}frac}$ ∧
  x $\otimes_{rec\text{-}rng\text{-}of\text{-}frac}$ *y* = $\mathbf{1}_{rec\text{-}rng\text{-}of\text{-}frac}$
  using *rec-rng-of-frac-def f1 f2 a2 rel-def a1*
  by (*metis* (*no-types*, *lifting*) *class-of-zero-rng-of-frac closed-add-rng-of-frac l-unit-add-rng-of-frac*

    mem-Sigma-iff monoid.select-convs(*2*) *partial-object.select-convs*(*1*) *semiring-simprules*(*4*) *zero-closed*)
have *x* ∈ *carrier rec-rng-of-frac*
  using *a2 a1 subset rev-subsetD rec-rng-of-frac-def*
  by (*metis* (*no-types*, *opaque-lifting*) *ring-hom-closed rng-to-rng-of-frac-def rng-to-rng-of-frac-is-ring-hom*)
thus *?thesis*
  using *Units-def*[*of rec-rng-of-frac*] *f3*
  by *auto*
qed

lemma *eq-class-to-rel*:
  assumes $(r, s) \in$ *carrier* $R \times S$ **and** $(r', s') \in$ *carrier* $R \times S$ **and** $(r \mid_{rel} s) = (r' \mid_{rel} s')$
  shows $(r, s) .=_{rel} (r', s')$
proof−
  have $(r, s) \in (r \mid_{rel} s)$
    using *assms*(*1*) *equiv-obj-rng-of-frac equivalence-def*
    by (*metis* (*no-types*, *lifting*) *CollectI case-prodI eq-class-of-rng-of-frac-def partial-object.select-convs*(*1*) *rel-def*)
  then have $(r, s) \in (r' \mid_{rel} s')$
    using *assms*(*3*)
    by *simp*
  then have $(r', s') .=_{rel} (r, s)$
    by (*simp add*: *eq-class-of-rng-of-frac-def*)
  thus *?thesis*
    using *equiv-obj-rng-of-frac equivalence-def*
    by (*metis* (*no-types*, *lifting*) *assms*(*1*) *assms*(*2*) *partial-object.select-convs*(*1*) *rel-def*)
qed

lemma *rng-to-rng-of-frac-without-zero-div-is-inj*:
  assumes $\mathbf{0} \notin S$ **and** $\forall a \in$ *carrier* $R.\forall b \in$ *carrier* $R.\ a \otimes b = \mathbf{0} \longrightarrow a = \mathbf{0} \vee b = \mathbf{0}$
  shows *a-kernel R rec-rng-of-frac rng-to-rng-of-frac* $= \{\mathbf{0}\}$
proof−
  have $\{r \in$ *carrier R. rng-to-rng-of-frac* $r = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}\} \subseteq \{\mathbf{0}\}$
  proof(*rule subsetI*)
    fix x
    assume *a1*:$x \in \{r \in$ *carrier R. rng-to-rng-of-frac* $r = \mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}\}$
    then have $(x, \mathbf{1}) .=_{rel} (\mathbf{0}, \mathbf{1})$
      using *rng-to-rng-of-frac-def rec-rng-of-frac-def eq-class-to-rel*
      by *simp*
    then obtain t **where** *f1*:$t \in S$ **and** *f2*:$t \otimes (\mathbf{1} \otimes x \ominus \mathbf{1} \otimes \mathbf{0}) = \mathbf{0}$
      using *rel-def*
      by *auto*
    have *f3*:$x \in$ *carrier R*
      using *a1*
      by *simp*
    then have *f4*:$t \otimes x = \mathbf{0}$
      using *l-one r-zero f2*
      by (*simp add*: *a-minus-def*)
    have $t \neq \mathbf{0}$
      using *f1 assms*(*1*)
      by *auto*
    then have $x = \mathbf{0}$
      using *assms*(*2*) *f1 f3 f4 subset rev-subsetD*
      by *auto*
    thus $x \in \{\mathbf{0}\}$
      by *simp*

qed
have {**0**} ⊆ {*r*∈ *carrier R*. *rng-to-rng-of-frac r* = $\mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$}
using *subsetI rng-to-rng-of-frac-def rec-rng-of-frac-def*
by *simp*
then have {*r*∈ *carrier R*. *rng-to-rng-of-frac r* = $\mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$} = {**0**}
using ‹{*r*∈ *carrier R*. *rng-to-rng-of-frac r* = $\mathbf{0}_{rec\text{-}rng\text{-}of\text{-}frac}$} ⊆ {**0**}›
by *auto*
thus *?thesis*
by (*simp add*: *a-kernel-def kernel-def*)
qed

end

end

2 Acknowledgements

The author was supported by the ERC Advanced Grant ALEXANDRIA (Project 742178) funded by the European Research Council and led by Professor Lawrence Paulson at the University of Cambridge, UK.

References

[1] S. Lang. *Algebra.* Springer, revised third edition edition, 2002.