archive-of-graph-formalizations

christoph

July 6, 2022

Contents

theory *defs*
  imports *Jordan-Normal-Form.Matrix*
    Jordan-Normal-Form.Gram-Schmidt
    LLL-Basis-Reduction.Gram-Schmidt-2
  Jordan-Normal-Form.DL-Missing-Sublist
Linear-Inequalities.Integer-Hull
HOL−Analysis.Determinants
HOL−Combinatorics.Permutations

begin

lemma *det-0-iff-vec-prod-zero1*: **assumes** A: $(A :: 'a :: idom\ Matrix.mat) \in carrier\text{-}mat\ n\ n$
  shows $Determinant.det\ A = 0 \longleftrightarrow (\exists\ v.\ v \in carrier\text{-}vec\ n \wedge v \neq 0_v\ n \wedge A *_v v = 0_v\ n)$
  using *det-0-iff-vec-prod-zero assms* **by** *auto*

definition
  $unit\text{-}vec1 :: nat \Rightarrow nat \Rightarrow ('b :: zero\text{-}neq\text{-}one)\ Matrix.vec$
  where $unit\text{-}vec1\ n\ i = Matrix.vec\ n\ (\lambda\ j.\ if\ j = i\ then\ 1\ else\ 0)$

no-notation *one-mat* $(\ 1_m)$

definition $one\text{-}mat1 :: nat \Rightarrow 'a :: \{zero,one\}\ Matrix.mat\ (1_m)$ **where**
  $1_m\ n \equiv Matrix.mat\ n\ n\ (\lambda\ (i,j).\ if\ i = j\ then\ 1\ else\ 0)$

proposition *cramer*:
  fixes $A ::'a::\{field\}\char`^'n\char`^'n$
  assumes *d0*: $det\ A \neq 0$
  shows $A *v\ x = b \longleftrightarrow x = (\chi\ k.\ det(\chi\ i\ j.\ if\ j{=}k\ then\ b\$i\ else\ A\$i\$j)\ /\ det\ A)$
proof −
  from *d0* **obtain** B **where** B: $A ** B = mat\ 1\ B ** A = mat\ 1$
    unfolding *invertible-det-nz*[*symmetric*] *invertible-def*
    by *blast*
  have $(A ** B) *v\ b = b$
    by (*simp add*: B)
  then have $A *v (B *v\ b) = b$
    by (*simp add*: *matrix-vector-mul-assoc*)

then have *xe*: $\exists x.\ A *v\ x = b$
  by *blast*
{
  fix *x*
  assume *x*: $A *v\ x = b$
  have $x = (\chi\ k.\ det(\chi\ i\ j.\ if\ j{=}k\ then\ b\$i\ else\ A\$i\$j)\ /\ det\ A)$
    unfolding *x*[*symmetric*]
    using *d0* **by** (*simp add*: *vec-eq-iff cramer-lemma field-simps*)
}
with *xe* **show** *?thesis*
  by *auto*
qed

no-notation *inner* (**infix** · *70*)
no-notation *Finite-Cartesian-Product.vec.vec-nth* (**infixl** \$ *90*)

context *gram-schmidt-floor*
begin

definition *mat-delete1 A i j* ≡
  $Matrix.mat\ (dim\text{-}row\ A - 1)\ (dim\text{-}col\ A - 1)\ (\lambda(i',j').$
    $A\ \$\$\ (if\ i' < i\ then\ i'\ else\ Suc\ i',\ if\ j' < j\ then\ j'\ else\ Suc\ j'))$

corollary *integer-hull-of-polyhedron1*: **assumes** *A*: $A \in carrier\text{-}mat\ nr\ n$
  and *b*: $b \in carrier\text{-}vec\ nr$
  and *AI*: $A \in \mathbb{Z}_m$
  and *bI*: $b \in \mathbb{Z}_v$
  and *P*: $P = polyhedron\ A\ b$
shows $\exists\ A'\ b'\ nr'.\ A' \in carrier\text{-}mat\ nr'\ n \wedge b' \in carrier\text{-}vec\ nr' \wedge integer\text{-}hull\ P = polyhedron\ A'\ b'$
proof −
  from *decomposition-theorem-integer-hull-of-polyhedron*[*OF A b AI bI P refl*]
  obtain *H C*
    where *HC*: $H \cup C \subseteq carrier\text{-}vec\ n \cap \mathbb{Z}_v$ *finite* $(H \cup C)$
      and *decomp*: $integer\text{-}hull\ P = convex\text{-}hull\ H + cone\ C$ **by** *auto*
  show *?thesis*
   by (*rule decomposition-theorem-polyhedra-2*[*OF - - - - decomp*], *insert HC*, *auto*)
qed
end

fun *pick* :: *nat set* ⇒ *nat* ⇒ *nat* **where**
$pick\ S\ 0 = (LEAST\ a.\ a{\in}S)\ |$
$pick\ S\ (Suc\ n) = (LEAST\ a.\ a{\in}S \wedge a > pick\ S\ n)$

lift-definition *dim-row* :: *'a mat* ⇒ *nat* **is** *fst* **.**
lift-definition *dim-col* :: *'a mat* ⇒ *nat* **is** *fst o snd* **.**
definition *carrier-mat* :: *nat* ⇒ *nat* ⇒ *'a mat set*
  where *carrier-mat nr nc* = { *m* . *dim-row m* = *nr* ∧ *dim-col m* = *nc*}

definition *undef-vec* :: *nat* ⇒ *'a* **where**
  undef-vec i ≡ [] ! *i*

definition *mk-vec* :: *nat* ⇒ (*nat* ⇒ *'a*) ⇒ (*nat* ⇒ *'a*) **where**
  mk-vec n f ≡ λ *i*. *if i* < *n then f i else undef-vec* (*i* − *n*)

typedef *'a vec* = {(*n*, *mk-vec n f*) | *n f* :: *nat* ⇒ *'a*. *True*}
  by *auto*

definition *mk-mat* :: *nat* ⇒ *nat* ⇒ (*nat* × *nat* ⇒ *'a*) ⇒ (*nat* × *nat* ⇒ *'a*) **where**
  mk-mat nr nc f ≡ λ (*i*,*j*). *if i* < *nr* ∧ *j* < *nc then f* (*i*,*j*) *else undef-mat nr nc f* (*i*,*j*)

lemma *cong-mk-mat*: **assumes** ⋀ *i j*. *i* < *nr* ⟹ *j* < *nc* ⟹ *f* (*i*,*j*) = *f'* (*i*,*j*)
  shows *mk-mat nr nc f* = *mk-mat nr nc f'*
  using *undef-cong-mat*[*of nr nc f f'*, *OF assms*]
  using *assms* **unfolding** *mk-mat-def*
  by *auto*

typedef *'a mat* = {(*nr*, *nc*, *mk-mat nr nc f*) | *nr nc f* :: *nat* × *nat* ⇒ *'a*. *True*}
  by *auto*

locale *gram-schmidt1* = *cof-vec-space n f-ty*
  for *n* :: *nat* **and** *f-ty* :: *'a* :: {*trivial-conjugatable-linordered-field*} *itself*
begin

definition *nonneg-lincomb c Vs b* = (*lincomb c Vs* = *b* ∧ *c* ' *Vs* ⊆ {*x*. *x* ≥ *0*})
definition *nonneg-lincomb-list c Vs b* = (*lincomb-list c Vs* = *b* ∧ (∀ *i* < *length Vs*. *c i* ≥ *0*))
definition *convex-lincomb c Vs b* = (*nonneg-lincomb c Vs b* ∧ *sum c Vs* = *1*)
definition *convex-lincomb-list c Vs b* = (*nonneg-lincomb-list c Vs b* ∧ *sum c* {*0*..<*length Vs*} = *1*)
definition *convex-hull Vs* = {*x*. ∃ *Ws c*. *finite Ws* ∧ *Ws* ⊆ *Vs* ∧ *convex-lincomb c Ws x*}
definition *convex S* = (*convex-hull S* = *S*)
definition *polyhedron A b* = {*x* ∈ *carrier-vec n*. $A *_v x \leq b$}

definition *integer-hull P* = *convex-hull* ($P \cap \mathbb{Z}_v$)
end

end