

# The Nash-Williams Theorem

Lawrence C. Paulson

March 24, 2023

## Abstract

In 1965, Nash-Williams [1] discovered a generalisation of the infinite form of Ramsey’s theorem. Where the latter concerns infinite sets of  $n$ -element sets for some fixed  $n$ , the Nash-Williams theorem concerns infinite sets of finite sets (or lists) subject to a “no initial segment” condition. The present formalisation follows Todorčević [2].

## Contents

<b>1</b>	<b>The Pointwise Less-Than Relation Between Two Sets</b>	<b>1</b>
<b>2</b>	<b>The Nash-Williams Theorem</b>	<b>2</b>
2.1	Initial segments . . . . .	3
2.2	Definitions and basic properties . . . . .	4
2.3	Main Theorem . . . . .	16
<b>3</b>	<b>Acknowledgements</b>	<b>18</b>

## 1 The Pointwise Less-Than Relation Between Two Sets

**theory** *Nash-Extras*

**imports** *HOL-Library.Ramsey* *HOL-Library.Countable-Set*

**begin**

**definition** *less-sets* :: [ $'a::\text{order set}$ ,  $'a::\text{order set}$ ]  $\Rightarrow$  *bool* (**infixr**  $\ll$  50)  
**where**  $A \ll B \equiv \forall x \in A. \forall y \in B. x < y$

**lemma** *less-sets-empty*[*iff*]:  $S \ll \{\} \ \{\} \ll T$   
**by** (*auto simp: less-sets-def*)

**lemma** *less-setsD*:  $\llbracket A \ll B; a \in A; b \in B \rrbracket \Longrightarrow a < b$   
**by** (*auto simp: less-sets-def*)

```

lemma less-sets-irrefl [simp]:  $A \ll A \longleftrightarrow A = \{\}$ 
  by (auto simp: less-sets-def)

lemma less-sets-trans:  $\llbracket A \ll B; B \ll C; B \neq \{\} \rrbracket \implies A \ll C$ 
  unfolding less-sets-def using less-trans by blast

lemma less-sets-weaken1:  $\llbracket A' \ll B; A \subseteq A' \rrbracket \implies A \ll B$ 
  by (auto simp: less-sets-def)

lemma less-sets-weaken2:  $\llbracket A \ll B'; B \subseteq B' \rrbracket \implies A \ll B$ 
  by (auto simp: less-sets-def)

lemma less-sets-imp-disjnt:  $A \ll B \implies \text{disjnt } A B$ 
  by (auto simp: less-sets-def disjnt-def)

lemma less-sets-UN1:  $\text{less-sets } (\bigcup \mathcal{A}) B \longleftrightarrow (\forall A \in \mathcal{A}. A \ll B)$ 
  by (auto simp: less-sets-def)

lemma less-sets-UN2:  $\text{less-sets } A (\bigcup \mathcal{B}) \longleftrightarrow (\forall B \in \mathcal{B}. A \ll B)$ 
  by (auto simp: less-sets-def)

lemma less-sets-Un1:  $\text{less-sets } (A \cup A') B \longleftrightarrow A \ll B \wedge A' \ll B$ 
  by (auto simp: less-sets-def)

lemma less-sets-Un2:  $\text{less-sets } A (B \cup B') \longleftrightarrow A \ll B \wedge A \ll B'$ 
  by (auto simp: less-sets-def)

lemma strict-sorted-imp-less-sets:
  strict-sorted (as @ bs)  $\implies (\text{list.set } \text{as}) \ll (\text{list.set } \text{bs})$ 
  by (simp add: less-sets-def sorted-wrt-append)

lemma Sup-nat-less-sets-singleton:
  fixes n::nat
  assumes Sup T < n finite T
  shows  $\text{less-sets } T \{n\}$ 
  using assms Max-less-iff
  by (auto simp: Sup-nat-def less-sets-def split: if-split-asm)

end

```

## 2 The Nash-Williams Theorem

Following S. Todorćević, *Introduction to Ramsey Spaces*, Princeton University Press (2010), 11–12.

```

theory Nash-Williams
  imports Nash-Extras
begin

```

**lemma** *finite-nat-Int-greaterThan-iff*:  
**fixes**  $N :: \text{nat set}$   
**shows**  $\text{finite } (N \cap \{n < ..\}) \longleftrightarrow \text{finite } N$   
**apply** (*simp add: finite-nat-iff-bounded subset-iff*)  
**by** (*metis dual-order.strict-trans2 nat-less-le not-less-eq*)

## 2.1 Initial segments

**definition** *init-segment*  $:: \text{nat set} \Rightarrow \text{nat set} \Rightarrow \text{bool}$   
**where**  $\text{init-segment } S T \equiv \exists S'. T = S \cup S' \wedge S \ll S'$

**lemma** *init-segment-subset*:  $\text{init-segment } S T \Longrightarrow S \subseteq T$   
**by** (*auto simp: init-segment-def*)

**lemma** *init-segment-refl*:  $\text{init-segment } S S$   
**by** (*metis empty-iff init-segment-def less-sets-def sup-bot.right-neutral*)

**lemma** *init-segment-antisym*:  $[\text{init-segment } S T; \text{init-segment } T S] \Longrightarrow S = T$   
**by** (*auto simp: init-segment-def*)

**lemma** *init-segment-trans*:  $[\text{init-segment } S T; \text{init-segment } T U] \Longrightarrow \text{init-segment } S U$

**unfolding** *init-segment-def*  
**by** (*meson UnE Un-assoc Un-upper1 less-sets-def less-sets-weaken1*)

**lemma** *init-segment-empty2 [iff]*:  $\text{init-segment } S \{\} \longleftrightarrow S = \{\}$   
**by** (*auto simp: init-segment-def less-sets-def*)

**lemma** *init-segment-Un*:  $S \ll S' \Longrightarrow \text{init-segment } S (S \cup S')$   
**by** (*auto simp: init-segment-def less-sets-def*)

**lemma** *init-segment-iff0*:  
**shows**  $\text{init-segment } S T \longleftrightarrow S \subseteq T \wedge S \ll (T - S)$   
**by** (*smt (verit) DiffD1 DiffD2 Diff-partition UnE init-segment-def init-segment-subset less-sets-def*)

**lemma** *init-segment-iff*:  
**shows**  $\text{init-segment } S T \longleftrightarrow S = T \vee (\exists m \in T. S = \{n \in T. n < m\})$  (*is ?lhs=?rhs*)

**proof**  
**assume** *?lhs*  
**then obtain**  $S'$  **where**  $S': T = S \cup S' \wedge S \ll S'$   
**by** (*meson init-segment-def*)  
**then have**  $S \subseteq T$   
**by** *auto*  
**then have**  $eq: S' = T - S$   
**using**  $S'$  **by** (*auto simp: less-sets-def*)  
**show** *?rhs*  
**proof** (*cases*  $T \subseteq S$ )

```

  case True
  with ⟨S ⊆ T⟩ show ?rhs by blast
next
  case False
  then have Inf S' ∈ T
    by (metis Diff-eq-empty-iff Diff-iff Inf-nat-def1 eq)
  moreover have ∧x. x ∈ S ⇒ x < Inf S'
    using S' False by (metis Diff-eq-empty-iff Inf-nat-def1 eq less-sets-def)
  moreover have {n ∈ T. n < Inf S'} ⊆ S
    using Inf-nat-def eq not-less-Least by fastforce
  ultimately show ?rhs
    using ⟨S ⊆ T⟩ by blast
qed
next
  assume ?rhs
  then show ?lhs
  proof (elim disjE bexE)
    fix m
    assume m: m ∈ T S = {n ∈ T. n < m}
    then have T = S ∪ {n ∈ T. m ≤ n}
      by auto
    moreover have S ≪ {n ∈ T. m ≤ n}
      using m by (auto simp: less-sets-def)
    ultimately show init-segment S T
      using init-segment-Un by force
  qed (use init-segment-refl in blast)
qed

```

**lemma** *init-segment-empty* [iff]: *init-segment* {} S  
 by (auto simp: *init-segment-def less-sets-def*)

**lemma** *init-segment-insert-iff*:  
 assumes Sn: S ≪ {n} and TS: ∧x. x ∈ T - S ⇒ n ≤ x  
 shows *init-segment* (insert n S) T ↔ *init-segment* S T ∧ n ∈ T  
 using DiffD1 Sn TS *init-segment-iff0 less-sets-def order-less-le* by fastforce

**lemma** *init-segment-insert*:  
 assumes *init-segment* S T and T: T ≪ {n}  
 shows *init-segment* S (insert n T)  
 by (metis assms *init-segment-Un init-segment-trans insert-is-Un sup-commute*)

## 2.2 Definitions and basic properties

**definition** *Ramsey* :: [nat set set, nat] ⇒ bool  
 where *Ramsey* F r ≡ ∀f ∈ F → {..
     ∀M. infinite M →  
       (∃N i. N ⊆ M ∧ infinite N ∧ i < r ∧  
         (∀j < r. j ≠ i → f - {j} ∩ F ∩ Pow N = {}))

Alternative, simpler definition suggested by a referee.

**lemma** *Ramsey-eq*:

*Ramsey*  $\mathcal{F} r \longleftrightarrow (\forall f \in \mathcal{F} \rightarrow \{..<r\}.$   
 $\forall M. \text{infinite } M \longrightarrow$   
 $(\exists N i. N \subseteq M \wedge \text{infinite } N \wedge i < r \wedge \mathcal{F} \cap \text{Pow } N \subseteq f -'$   
 $\{i\}))$

**unfolding** *Ramsey-def*

**by** (*intro ball-cong all-cong ex-cong1 conj-cong refl*) *blast*

**definition** *thin-set* :: *nat set set*  $\Rightarrow$  *bool*

**where** *thin-set*  $\mathcal{F} \equiv \mathcal{F} \subseteq \text{Collect finite} \wedge (\forall S \in \mathcal{F}. \forall T \in \mathcal{F}. \text{init-segment } S T \longrightarrow S = T)$

**definition** *comparables* :: *nat set*  $\Rightarrow$  *nat set*  $\Rightarrow$  *nat set set*

**where** *comparables*  $S M \equiv \{T. \text{finite } T \wedge (\text{init-segment } T S \vee \text{init-segment } S T \wedge T - S \subseteq M)\}$

**lemma** *comparables-iff*:  $T \in \text{comparables } S M \longleftrightarrow \text{finite } T \wedge (\text{init-segment } T S \vee \text{init-segment } S T \wedge T \subseteq S \cup M)$

**by** (*auto simp: comparables-def init-segment-def*)

**lemma** *comparables-subset*:  $\bigcup (\text{comparables } S M) \subseteq S \cup M$

**by** (*auto simp: comparables-def init-segment-def*)

**lemma** *comparables-empty* [*simp*]: *comparables*  $\{\}$   $M = \text{Fpow } M$

**by** (*auto simp: comparables-def Fpow-def*)

**lemma** *comparables-mono*:  $N \subseteq M \Longrightarrow \text{comparables } S N \subseteq \text{comparables } S M$

**by** (*auto simp: comparables-def*)

**definition** *rejects*  $\mathcal{F} S M \equiv \text{comparables } S M \cap \mathcal{F} = \{\}$

**abbreviation** *accepts*

**where** *accepts*  $\mathcal{F} S M \equiv \neg \text{rejects } \mathcal{F} S M$

**definition** *strongly-accepts*

**where** *strongly-accepts*  $\mathcal{F} S M \equiv (\forall N \subseteq M. \text{rejects } \mathcal{F} S N \longrightarrow \text{finite } N)$

**definition** *decides*

**where** *decides*  $\mathcal{F} S M \equiv \text{rejects } \mathcal{F} S M \vee \text{strongly-accepts } \mathcal{F} S M$

**definition** *decides-subsets*

**where** *decides-subsets*  $\mathcal{F} M \equiv \forall T. T \subseteq M \longrightarrow \text{finite } T \longrightarrow \text{decides } \mathcal{F} T M$

**lemma** *strongly-accepts-imp-accepts*:

$\llbracket \text{strongly-accepts } \mathcal{F} S M; \text{infinite } M \rrbracket \Longrightarrow \text{accepts } \mathcal{F} S M$

**unfolding** *strongly-accepts-def* **by** *blast*

**lemma** *rejects-trivial*:  $\llbracket \text{rejects } \mathcal{F} S M; \text{thin-set } \mathcal{F}; \text{init-segment } F S; F \in \mathcal{F} \rrbracket \Longrightarrow$

*False*

**unfolding** *rejects-def thin-set-def*  
**using** *comparables-iff* **by** *blast*

**lemma** *rejects-subset*:  $\llbracket \text{rejects } \mathcal{F} S M; N \subseteq M \rrbracket \implies \text{rejects } \mathcal{F} S N$   
**by** (*fastforce simp add: rejects-def comparables-def*)

**lemma** *strongly-accepts-subset*:  $\llbracket \text{strongly-accepts } \mathcal{F} S M; N \subseteq M \rrbracket \implies \text{strongly-accepts } \mathcal{F} S N$   
**by** (*auto simp: strongly-accepts-def*)

**lemma** *decides-subset*:  $\llbracket \text{decides } \mathcal{F} S M; N \subseteq M \rrbracket \implies \text{decides } \mathcal{F} S N$   
**by** (*meson decides-def rejects-subset strongly-accepts-subset*)

**lemma** *decides-subsets-subset*:  $\llbracket \text{decides-subsets } \mathcal{F} M; N \subseteq M \rrbracket \implies \text{decides-subsets } \mathcal{F} N$   
**by** (*meson decides-subset decides-subsets-def subset-trans*)

**lemma** *rejects-empty [simp]*:  $\text{rejects } \mathcal{F} \{ \} M \longleftrightarrow \text{Fpow } M \cap \mathcal{F} = \{ \}$   
**by** (*auto simp: rejects-def comparables-def Fpow-def*)

**lemma** *strongly-accepts-empty [simp]*:  $\text{strongly-accepts } \mathcal{F} \{ \} M \longleftrightarrow (\forall N \subseteq M. \text{Fpow } N \cap \mathcal{F} = \{ \} \longrightarrow \text{finite } N)$   
**by** (*simp add: strongly-accepts-def Fpow-def disjoint-iff*)

**lemma** *ex-infinite-decides-1*:  
**assumes** *infinite M*  
**obtains** *N* **where**  $N \subseteq M$  *infinite N* *decides } \mathcal{F} S N  
**by** (*meson assms decides-def order-refl strongly-accepts-def*)*

**proposition** *ex-infinite-decides-finite*:

**assumes** *infinite M finite S*

**obtains** *N* **where**  $N \subseteq M$  *infinite N*  $\wedge T. T \subseteq S \implies \text{decides } \mathcal{F} T N$

**proof** –

**have** *finite (Pow S)*

**by** (*simp add: <finite S>*)

**then obtain** *f* **::**  $\text{nat} \Rightarrow \text{nat set}$  **where**  $f: f \text{ ' } \{ .. < \text{card } (\text{Pow } S) \} = \text{Pow } S$

**by** (*metis bij-betw-imp-surj-on [OF bij-betw-from-nat-into-finite]*)

**obtain** *M0* **where**  $M0: \text{infinite } M0$   $M0 \subseteq M$  *decides } \mathcal{F} (f 0) M0*

**by** (*meson <infinite M> ex-infinite-decides-1*)

**define** *F* **where**  $F \equiv \text{rec-nat } M0 (\lambda n N. @N'. N' \subseteq N \wedge \text{infinite } N' \wedge \text{decides } \mathcal{F} (f (Suc n)) N')$

**define**  $\Phi$  **where**  $\Phi \equiv \lambda n N'. N' \subseteq F n \wedge \text{infinite } N' \wedge \text{decides } \mathcal{F} (f (Suc n)) N'$

**have** *P-Suc*:  $F (Suc n) = (@N'. \Phi n N')$  **for** *n*

**by** (*auto simp: F-def } \Phi-def*)

**have**  $*$ :  $\text{infinite } (F n) \wedge \text{decides } \mathcal{F} (f n) (F n) \wedge F n \subseteq M$  **for** *n*

**proof** (*induction n*)

**case** (*Suc n*) **then show** *?case*

**by** (*metis P-Suc } \Phi-def ex-infinite-decides-1 someI-ex subset-trans*)

```

qed (auto simp: F-def M0)
then have telescope:  $F (Suc\ n) \subseteq F\ n$  for  $n$ 
  by (metis P-Suc  $\Phi$ -def ex-infinite-decides-1 someI-ex)
let  $?N = \bigcap_{n < card\ (Pow\ S)} F\ n$ 
show thesis
proof
  show  $?N \subseteq M$ 
    by (metis * INF-lower2 Pow-iff f imageE order-refl)
next
  have eq:  $(\bigcap_{n < Suc\ m} F\ n) = F\ m$  for  $m$ 
    by (induction m) (use telescope in <auto simp: lessThan-Suc>)
  then show infinite  $?N$ 
    by (metis (full-types) * Pow-top Suc-le-D Suc-le-eq f imageE lessThan-iff)
next
fix T
  assume  $T \subseteq S$  then show decides  $\mathcal{F}\ T\ ?N$ 
    by (metis (no-types) * INT-lower Pow-iff decides-subset f imageE)
qed
qed

```

**lemma** *sorted-wrt-subset*:  $\llbracket X \in list.set\ l; sorted-wrt\ (\leq)\ l \rrbracket \implies hd\ l \subseteq X$   
 by (induction l) auto

Todorčević's Lemma 1.18

**proposition** *ex-infinite-decides-subsets*:

assumes *thin-set*  $\mathcal{F}$  infinite  $M$

obtains  $N$  where  $N \subseteq M$  infinite  $N$  decides-subsets  $\mathcal{F}\ N$

**proof** –

obtain  $M0$  where  $M0$ : infinite  $M0$   $M0 \subseteq M$  decides  $\mathcal{F}\ \{\}\ M0$

by (meson <infinite  $M$ > ex-infinite-decides-1)

define *decides-all* where *decides-all*  $\equiv \lambda S\ N. \forall T \subseteq S. decides\ \mathcal{F}\ T\ N$

define  $\Phi$  where  $\Phi \equiv \lambda NL\ N. N \subseteq hd\ NL \wedge Inf\ N > Inf\ (hd\ NL) \wedge infinite\ N$   
 $\wedge decides-all\ (List.set\ (map\ Inf\ NL))\ N$

have  $\exists N. \Phi\ NL\ N$  if infinite (hd NL) for NL

**proof** –

obtain  $N$  where  $N$ :  $N \subseteq hd\ NL$  infinite  $N$  decides-all (List.set (map Inf NL))

$N$

unfolding *decides-all-def*

by (metis List.finite-set ex-infinite-decides-finite <infinite (hd NL)>)

then have *inf*: infinite  $(N \cap \{Inf\ (hd\ NL) <..\})$

by (metis finite-nat-Int-greaterThan-iff)

then have *Inf*  $(N \cap \{Inf\ (hd\ NL) <..\}) > Inf\ (hd\ NL)$

by (metis finite.emptyI Inf-nat-def1 Int-iff greaterThan-iff)

with  $N$  show *?thesis*

unfolding  $\Phi$ -def

by (meson Int-lower1 decides-all-def decides-subset inf subset-trans)

qed

then have  $\Phi$ -Eps:  $\Phi\ NL\ (Eps\ (\Phi\ NL))$  if infinite (hd NL) for NL

```

  by (simp add: someI-ex that)
define  $F$  where  $F \equiv \text{rec-nat } [M0] (\lambda n \text{ NL. } (Eps (\Phi \text{ NL})) \# \text{ NL})$ 
have  $F\text{-simps } [simp]: F \ 0 = [M0] \ F \ (Suc \ n) = Eps (\Phi (F \ n)) \# \ F \ n$  for  $n$ 
  by (auto simp:  $F\text{-def}$ )
have  $F: F \ n \neq [] \wedge \text{sorted-wrt } (\leq) (F \ n) \wedge \text{list.set } (F \ n) \subseteq \text{Collect infinite} \wedge$ 
 $\text{list.set } (F \ n) \subseteq \text{Pow } M$  for  $n$ 
proof (induction  $n$ )
  case  $0$ 
  then show ?case
    by (simp add:  $M0$ )
next
  case ( $Suc \ n$ )
  then have  $*$ :  $\Phi (F \ n) (Eps (\Phi (F \ n)))$ 
    using  $\Phi\text{-Eps hd-in-set}$  by blast
  show ?case
proof (intro conjI)
  show  $\text{sorted-wrt } (\subseteq) (F (Suc \ n))$ 
    using subset-trans [ $OF - \text{sorted-wrt-subset}$ ]  $Suc.IH \ \Phi\text{-def} \ *$  by auto
  show  $\text{list.set } (F (Suc \ n)) \subseteq \{S. \text{infinite } S\}$ 
    using  $*$   $\Phi\text{-def} \ Suc.IH$  by force
  show  $\text{list.set } (F (Suc \ n)) \subseteq \text{Pow } M$ 
    using  $*$   $Suc.IH \ \Phi\text{-def hd-in-set}$  by force
  qed auto
qed
have  $\Phi F: \Phi (F \ n) (Eps (\Phi (F \ n)))$  for  $n$ 
  using  $F \ \Phi\text{-Eps hd-in-set}$  by blast
then have  $\text{decides: decides-all } (\text{List.set } (\text{map } \text{Inf } (F \ n))) (Eps (\Phi (F \ n)))$  for  $n$ 
  using  $\Phi\text{-def}$  by blast
have  $\text{Eps-subset-hd: } Eps (\Phi (F \ n)) \subseteq \text{hd } (F \ n)$  for  $n$ 
  using  $\Phi F \ \Phi\text{-def}$  by blast
have  $\text{List.set } (\text{map } \text{Inf } (F \ n)) \subseteq \text{List.set } (\text{map } \text{Inf } (F (Suc \ n)))$  for  $n$ 
  by auto
then have  $\text{map-Inf-subset: } m \leq n \implies \text{List.set } (\text{map } \text{Inf } (F \ m)) \subseteq \text{List.set } (\text{map } \text{Inf } (F \ n))$  for  $m \ n$ 
  by (rule order-class.lift-Suc-mono-le) auto
define  $mmap$  where  $mmap \equiv \lambda n. \text{Inf } (\text{hd } (F (Suc \ n)))$ 
have  $mmap \ n < mmap (Suc \ n)$  for  $n$ 
  by (metis  $F\text{-simps}(2) \ \Phi F \ \Phi\text{-def list.sel}(1) \ mmap\text{-def}$ )
then have  $\text{strict-mono } mmap$ 
  by (simp add: lift-Suc-mono-less strict-mono-def)
have  $\text{finite-F-bound: } \exists n. S \subseteq \text{List.set } (\text{map } \text{Inf } (F \ n))$ 
  if  $S: S \subseteq \text{range } mmap \ \text{finite } S$  for  $S$ 
proof –
  obtain  $K$  where  $\text{finite } K \ S \subseteq mmap \ 'K$ 
  by (metis  $S \ \text{finite-subset-image order-refl}$ )
  show ?thesis
proof
  have  $mmap \ 'K \subseteq \text{list.set } (\text{map } \text{Inf } (F (Suc (Max \ K))))$ 
    unfolding  $mmap\text{-def image-subset-iff}$ 

```



```

    by (metis F Max-ge Suc-le-mono ⟨finite K⟩ hd-in-set imageI map-Inf-subset
set-map subsetD)
    with S show S ⊆ list.set (map Inf (F (Suc (Max K))))
      using ⟨S ⊆ mmap ‘K’⟩ by auto
    qed
  qed
  have Eps (Φ (F (Suc n))) ⊆ Eps (Φ (F n)) for n
    by (metis F-simps(2) ΦF Φ-def list.sel(1))
  then have Eps-Φ-decreasing: m ≤ n ⇒ Eps (Φ (F n)) ⊆ Eps (Φ (F m)) for
m n
    by (rule order-class.lift-Suc-antimono-le)
  have hd-Suc-eq-Eps: hd (F (Suc n)) = Eps (Φ (F n)) for n
    by simp
  have Inf (hd (F n)) ∈ hd (F n) for n
    by (metis Inf-nat-def1 ΦF Φ-def finite.emptyI rev-finite-subset)
  then have Inf-hd-in-Eps: Inf (hd (F m)) ∈ Eps (Φ (F n)) if m > n for m n
    by (metis Eps-Φ-decreasing Nat.lessE hd-Suc-eq-Eps less-imp-le-nat subsetD
that)
  then have image-mmap-subset-hd-F: mmap ‘{n..}’ ⊆ hd (F (Suc n)) for n
    by (metis hd-Suc-eq-Eps atLeast-iff image-subsetI le-imp-less-Suc mmap-def)
  have list.set (F k) ⊆ list.set (F n) if k ≤ n for k n
    by (rule order-class.lift-Suc-mono-le) (use that in auto)
  then have hd-F-in-F: hd (F k) ∈ list.set (F n) if k ≤ n for k n
    by (meson F hd-in-set subsetD that)
  show thesis
proof
  show infinite-mm: infinite (range mmap)
    using ⟨strict-mono mmap⟩ finite-imageD strict-mono-on-imp-inj-on by blast
  show range mmap ⊆ M
    using Eps-subset-hd ⟨M0 ⊆ M⟩ image-mmap-subset-hd-F by fastforce
  show decides-subsets ℱ (range mmap)
    unfolding decides-subsets-def
  proof (intro strip)
    fix S
    assume S ⊆ range mmap finite S
    define n where n ≡ LEAST n. S ⊆ List.set (map Inf (F n))
    have ∃ m. S ⊆ List.set (map Inf (F m))
      using ⟨S ⊆ range mmap⟩ ⟨finite S⟩ finite-F-bound by blast
    then have S: S ⊆ List.set (map Inf (F n)) and minS: ⋀ m. m < n ⇒ ¬ S
⊆ List.set (map Inf (F m))
      unfolding n-def by (meson LeastI-ex not-less-Least)+
    have decides-Fn: decides ℱ S (Eps (Φ (F n)))
      using S decides decides-all-def by blast
    show decides ℱ S (range mmap)
  proof (cases n=0)
    case True
    then show ?thesis
      by (metis image-mmap-subset-hd-F decides-Fn decides-subset hd-Suc-eq-Eps
atLeast-0)
  end
end

```

```

next
  case False
  have notin-map-Inf:  $x \notin \text{List.set (map Inf (F n))}$  if  $S \ll \{x\}$  for  $x$ 
  proof clarsimp
    fix  $T$ 
    assume  $x = \text{Inf } T$  and  $T \in \text{list.set (F n)}$ 
    with that have ls:  $S \ll \{\text{Inf } T\}$ 
      by auto
    have  $S \subseteq \text{List.set (map Inf (F j))}$  if  $T: T \in \text{list.set (F (Suc j))}$  for  $j$ 
    proof clarsimp
      fix  $x$ 
      assume  $x \in S$ 
      then have  $x < \text{Inf } T$ 
        using less-setsD ls by blast
      then have  $x \notin T$ 
        using Inf-nat-def not-less-Least by auto
      obtain  $k$  where  $k: x = \text{mmap } k$ 
        using  $\langle S \subseteq \text{range mmap} \rangle \langle x \in S \rangle$  by blast
      moreover have  $\text{Eps } (\Phi (F j)) \subseteq T$ 
        by (metis F hd-Suc-eq-Eps sorted-wrt-subset that)
      ultimately have  $k < j$ 
        unfolding mmap-def by (metis Inf-hd-in-Eps  $\langle x \notin T \rangle$  in-mono
not-less-eq)
      then show  $x \in \text{Inf } \text{'list.set (F j)}$ 
        using Suc-leI hd-F-in-F k mmap-def by blast
    qed
  then show False
    by (metis False  $\langle T \in \text{set (F n)} \rangle$  lessI minS not0-implies-Suc)
  qed
  have Inf-hd-F:  $\text{Inf (hd (F m))} \in \text{Eps } (\Phi (F n))$  if  $S \ll \{\text{Inf (hd (F m))}\}$ 
  for  $m$ 
    by (metis Inf-hd-in-Eps hd-F-in-F notin-map-Inf imageI leI set-map that)
  have less-S:  $S \ll \{\text{Inf (hd (F m))}\}$ 
    if init-segment S T  $\text{Inf (hd (F m))} \in T$   $\text{Inf (hd (F m))} \notin S$  for  $T m$ 
    using  $\langle \text{finite } S \rangle$  that by (auto simp: init-segment-iff less-sets-def)
  consider rejects  $\mathcal{F} S (\text{Eps } (\Phi (F n))) \mid \text{strongly-accepts } \mathcal{F} S (\text{Eps } (\Phi (F$ 
n))))
    using decides-Fn by (auto simp: decides-def)
  then show ?thesis
  proof cases
    case 1
      then have rejects  $\mathcal{F} S (\text{range mmap})$ 
      apply (simp add: rejects-def disjoint-iff mmap-def comparables-def image-iff
subset-iff)
        by (metis less-S Inf-hd-F hd-Suc-eq-Eps)
      then show ?thesis
        by (auto simp: decides-def)
    next
      case 2

```

```

have False
  if  $N \subseteq \text{range } \text{mmap } \text{and } \text{rejects } \mathcal{F} S N \text{ and } \text{infinite } N \text{ for } N$ 
proof –
  have  $N = \text{mmap } \{n..\} \cap N \cup \text{mmap } \{..<n\} \cap N$ 
    using in-mono that(1) by fastforce
  then have infinite ( $\text{mmap } \{n..\} \cap N$ )
    by (metis finite-Int finite-Un finite-imageI finite-lessThan <infinite N>)
  moreover have rejects  $\mathcal{F} S$  ( $\text{mmap } \{n..\} \cap N$ )
    using rejects-subset <rejects  $\mathcal{F} S N$  > by fastforce
  moreover have  $\text{mmap } \{n..\} \cap N \subseteq \text{Eps } (\Phi (F n))$ 
    using image-mmap-subset-hd-F by fastforce
  ultimately show ?thesis
    using 2 by (auto simp: strongly-accepts-def)
qed
with 2 show ?thesis
  by (auto simp: decides-def strongly-accepts-def)
qed
qed
qed
qed
qed

```

Todorčević's Lemma 1.19

```

proposition strongly-accepts-1-19:
  assumes acc: strongly-accepts  $\mathcal{F} S M$ 
  and thin-set  $\mathcal{F}$  infinite  $M S \subseteq M$  finite  $S$ 
  and dsM: decides-subsets  $\mathcal{F} M$ 
  shows finite  $\{n \in M. \neg \text{strongly-accepts } \mathcal{F} (\text{insert } n S) M\}$ 
proof (rule ccontr)
  define  $N$  where  $N \equiv \{n \in M. \text{rejects } \mathcal{F} (\text{insert } n S) M\} \cap \{\text{Sup } S <..\}$ 
  have  $N \subseteq M$  and  $N: \bigwedge n. n \in N \longleftrightarrow n \in M \wedge \text{rejects } \mathcal{F} (\text{insert } n S) M \wedge n > \text{Sup } S$ 
    by (auto simp: N-def)
  assume  $\neg ?thesis$ 
  moreover have  $\{n \in M. \neg \text{strongly-accepts } \mathcal{F} (\text{insert } n S) M\} = \{n \in M. \text{rejects } \mathcal{F} (\text{insert } n S) M\}$ 
    using dsM <finite S> <infinite M> <S ⊆ M> unfolding decides-subsets-def
    by (meson decides-def finite.insertI insert-subset strongly-accepts-imp-accepts)
  ultimately have infinite  $N$ 
    by (simp add: N-def finite-nat-Int-greaterThan-iff)
  then have accepts  $\mathcal{F} S N$ 
    using acc strongly-accepts-def <N ⊆ M> by blast
  then obtain  $T$  where  $T: T \in \text{comparables } S N T \in \mathcal{F}$  and  $TSN: T \subseteq S \cup N$ 
    unfolding rejects-def using comparables-iff init-segment-subset by fastforce
  then consider init-segment  $T S \mid$  init-segment  $S T S \neq T \neg$  init-segment  $T S$ 
    by (auto simp: comparables-def)
  then show False
proof cases
  case 1

```

**then have** *init-segment*  $T$  (*insert*  $n$   $S$ ) **if**  $n \in N$  **for**  $n$   
 by (*meson* *Sup-nat-less-sets-singleton*  $N$   $\langle$ finite  $S$  $\rangle$  *init-segment-insert that*)  
**with**  $\langle$ infinite  $N$  $\rangle$   $\langle$ thin-set  $\mathcal{F}$  $\rangle$   $\langle T \in \mathcal{F} \rangle$  **show** *False*  
 by (*meson*  $N$  *infinite-nat-iff-unbounded rejects-trivial*)  
**next**  
**let**  $?n = \text{Min } (T - S)$   
**case**  $\mathcal{Q}$   
**then have**  $TS: \text{finite } (T - S) \ T - S \neq \{\}$   
 using  $T(1)$  *init-segment-subset* **by** (*force simp: comparables-iff*)  
**then have**  $?n \in N$   
 by (*meson* *Diff-subset-conv Min-in TSN subsetD*)  
**then have** *rejects*  $\mathcal{F}$  (*insert*  $?n$   $S$ )  $N$   
 using *rejects-subset*  $\langle N \subseteq M \rangle$  **by** (*auto simp: N-def*)  
**then have**  $\neg$  *init-segment*  $T$  (*insert*  $?n$   $S$ ) (*init-segment* (*insert*  $?n$   $S$ )  $T \longrightarrow$   
*insert*  $?n$   $S = T$ )  
 using  $T$  *Diff-partition TSN*  $\langle ?n \in N \rangle$   $\langle$ finite  $S$  $\rangle$   
 by (*auto simp: rejects-def comparables-iff disjoint-iff*)  
**moreover have**  $S \ll \{?n\}$   
 using *Sup-nat-less-sets-singleton*  $N$   $\langle ?n \in N \rangle$   $\langle$ finite  $S$  $\rangle$  **by** *blast*  
**ultimately show** *?thesis*  
 using  $\mathcal{Q}$  **by** (*metis* *DiffD1 eq-Min-iff TS init-segment-insert-iff*)  
**qed**  
**qed**

Much work is needed for this slight strengthening of the previous result!

**proposition** *strongly-accepts-1-19-plus:*

**assumes** *thin-set*  $\mathcal{F}$  *infinite*  $M$

**and** *dsM: decides-subsets*  $\mathcal{F}$   $M$

**obtains**  $N$  **where**  $N \subseteq M$  *infinite*  $N$

$\bigwedge S n. \llbracket S \subseteq N; \text{finite } S; \text{strongly-accepts } \mathcal{F} \ S \ N; n \in N; S \ll \{n\} \rrbracket$   
 $\implies \text{strongly-accepts } \mathcal{F} \ (\text{insert } n \ S) \ N$

**proof** –

**define** *insert-closed* **where**

*insert-closed*  $\equiv \lambda NL \ N. \forall T \subseteq \text{Inf } ' \text{set } NL. \forall n \in N.$

*strongly-accepts*  $\mathcal{F}$   $T \ ((\text{Inf } ' \text{set } NL) \cup \text{hd } NL) \longrightarrow$

$T \ll \{n\} \longrightarrow \text{strongly-accepts } \mathcal{F} \ (\text{insert } n \ T) \ ((\text{Inf } ' \text{set } NL) \cup$

*hd*  $NL)$

**define**  $\Phi$  **where**  $\Phi \equiv \lambda NL \ N. N \subseteq \text{hd } NL \wedge \text{Inf } N > \text{Inf } (\text{hd } NL) \wedge \text{infinite } N$

$\wedge$  *insert-closed*  $NL \ N$

**have**  $\exists N. \Phi \ NL \ N$  **if**  $NL: \text{infinite } (\text{hd } NL) \ \text{Inf } ' \ \text{set } NL \cup \text{hd } NL \subseteq M$  **for**  $NL$

**proof** –

**let**  $?m = \text{Inf } ' \ \text{set } NL$

**let**  $?M = ?m \cup \text{hd } NL$

**define**  $P$  **where**  $P \equiv \lambda S. \{n \in ?M. \neg \text{strongly-accepts } \mathcal{F} \ (\text{insert } n \ S) \ ?M\}$

**have**  $\exists k. P \ S \subseteq \{..k\}$

**if**  $S \subseteq \text{Inf } ' \ \text{set } NL$  *strongly-accepts*  $\mathcal{F}$   $S$   $?M$  **for**  $S$

**proof** –

**have** *decides-subsets*  $\mathcal{F}$   $?M$

**using**  $NL(2)$  *decides-subsets-subset*  $dsM$  **by** *blast*

```

with that NL assms finite-surj have finite (P S)
  unfolding P-def by (blast intro!: strongly-accepts-1-19)
then show ?thesis
  by (simp add: finite-nat-iff-bounded-le)
qed
then obtain f where f:  $\bigwedge S. \llbracket S \subseteq \text{Inf } ' \text{ set NL}; \text{ strongly-accepts } \mathcal{F} S ?M \rrbracket \implies$ 
P S  $\subseteq \{..f S\}$ 
  by metis
define m where m  $\equiv \text{Max } (\text{insert } (\text{Inf } (\text{hd } \text{NL})) (f ' \text{Pow } (\text{Inf } ' \text{ set NL})))$ 
have  $\S$ : strongly-accepts  $\mathcal{F}$  (insert n S) ?M
  if S: S  $\subseteq \text{Inf } ' \text{ set NL}$  strongly-accepts  $\mathcal{F} S ?M$  and n: n  $\in \text{hd } \text{NL} \cap \{m<..\}$ 
for S n
proof -
  have f S  $\leq m$ 
    unfolding m-def using that(1) by auto
  then show ?thesis
    using f [OF S] n unfolding P-def by auto
qed
have  $\Phi$  NL (hd NL  $\cap \{m<..\}$ )
  unfolding  $\Phi$ -def
proof (intro conjI)
  show infinite (hd NL  $\cap \{m<..\}$ )
    by (simp add: finite-nat-Int-greaterThan-iff that(1))
  moreover have Inf (hd NL)  $\leq m$ 
    by (simp add: m-def)
  ultimately show Inf (hd NL) < Inf (hd NL  $\cap \{m<..\}$ )
    using Inf-nat-def1 [of (hd NL  $\cap \{m<..\}$ )] by force
  show insert-closed NL (hd NL  $\cap \{m<..\}$ )
    by (auto intro:  $\S$  simp: insert-closed-def)
qed auto
then show ?thesis ..
qed
then have  $\Phi$ -Eps:  $\Phi$  NL (Eps ( $\Phi$  NL)) if infinite (hd NL) (Inf ' set NL)  $\cup$  hd
NL  $\subseteq M$  for NL
  by (meson someI-ex that)
define F where F  $\equiv \text{rec-nat } [M] (\lambda n \text{ NL}. (\text{Eps } (\Phi \text{ NL})) \# \text{NL})$ 
have F-simps [simp]: F 0 = [M] F (Suc n) = Eps ( $\Phi$  (F n)) # F n for n
  by (auto simp: F-def)
have InfM: Inf M  $\in M$ 
  by (metis Inf-nat-def1 assms(2) finite.emptyI)
have F: F n  $\neq [] \wedge \text{sorted-wrt } (\leq) (F n) \wedge \text{list.set } (F n) \subseteq \text{Collect infinite } \wedge \text{set}$ 
(F n)  $\subseteq \text{Pow } M \wedge \text{Inf } ' \text{ set } (F n) \subseteq M$  for n
proof (induction n)
  case (Suc n)
  have hd (F n)  $\subseteq M$ 
    by (meson Pow-iff Suc.IH hd-in-set subsetD)
  then obtain  $\Phi$ : Ball (list.set (F n)) (( $\subseteq$ ) (Eps ( $\Phi$  (F n)))) infinite (Eps ( $\Phi$  (F
n)))
    using order-trans [OF - sorted-wrt-subset]

```

by (*metis Suc.IH Un-subset-iff  $\Phi$ -Eps  $\Phi$ -def hd-in-set mem-Collect-eq subsetD*)  
 then have  $M: Eps (\Phi (F n)) \subseteq M$   
 by (*meson Pow-iff Suc.IH hd-in-set subset-iff*)  
 with  $\Phi$  have  $Inf (Eps (\Phi (F n))) \in M$   
 by (*metis Inf-nat-def1 finite.simps in-mono*)  
 with  $\Phi M$  show ?case  
 using *Suc by simp*  
 qed (*auto simp: InfM <infinite M>*)  
 have  $\Phi F: \Phi (F n) (Eps (\Phi (F n)))$  for  $n$   
 by (*metis Ball-Collect F Pow-iff Un-subset-iff  $\Phi$ -Eps hd-in-set subsetD*)  
 then have *insert-closed: insert-closed (F n) (Eps ( $\Phi (F n)$ ))* for  $n$   
 using  $\Phi$ -def by blast  
 have *Eps-subset-hd: Eps ( $\Phi (F n)) \subseteq hd (F n)$*  for  $n$   
 using  $\Phi F$   $\Phi$ -def by blast  
 define *mmap* where  $mmap \equiv \lambda n. Inf (hd (F (Suc n)))$   
 have  $mmap n < mmap (Suc n)$  for  $n$   
 by (*metis F-simps(2)  $\Phi F$   $\Phi$ -def list.sel(1) mmap-def*)  
 then have *strict-mono mmap*  
 by (*simp add: lift-Suc-mono-less strict-mono-def*)  
 then have *inj mmap*  
 by (*simp add: strict-mono-imp-inj-on*)  
 have  $Eps (\Phi (F (Suc n))) \subseteq Eps (\Phi (F n))$  for  $n$   
 by (*metis F-simps(2)  $\Phi F$   $\Phi$ -def list.sel(1)*)  
 then have *Eps- $\Phi$ -decreasing:  $m \leq n \implies Eps (\Phi (F n)) \subseteq Eps (\Phi (F m))$*  for  
 $m n$   
 by (*rule order-class.lift-Suc-antimono-le*)  
 have *hd-Suc-eq-Eps: hd (F (Suc n)) = Eps ( $\Phi (F n)$ )* for  $n$   
 by *simp*  
 have  $Inf (hd (F n)) \in hd (F n)$  for  $n$   
 by (*metis Inf-nat-def1  $\Phi F$   $\Phi$ -def finite.emptyI finite-subset*)  
 then have *Inf-hd-in-Eps: Inf (hd (F m))  $\in Eps (\Phi (F n))$  if  $m > n$*  for  $m n$   
 by (*metis Eps- $\Phi$ -decreasing Nat.lessE hd-Suc-eq-Eps nat-less-le subsetD that*)  
 then have *image-mmap-subset-hd-F:  $mmap \{n..\} \subseteq hd (F (Suc n))$*  for  $n$   
 by (*metis hd-Suc-eq-Eps atLeast-iff image-subsetI le-imp-less-Suc mmap-def*)  
 have  $list.set (F k) \subseteq list.set (F n)$  if  $k \leq n$  for  $k n$   
 by (*rule order-class.lift-Suc-mono-le*) (*use that in auto*)  
 then have *hd-F-in-F: hd (F k)  $\in list.set (F n)$  if  $k \leq n$*  for  $k n$   
 by (*meson F hd-in-set subsetD that*)  
 show ?thesis  
 proof  
 show *infinite-mm: infinite (range mmap)*  
 using *<inj mmap> range-inj-infinite by blast*  
 show  $range mmap \subseteq M$   
 using *Eps-subset-hd image-mmap-subset-hd-F by fastforce*  
 next  
 fix  $S a$   
 assume  $S: S \subseteq range mmap$  *finite S* and *acc: strongly-accepts  $\mathcal{F} S (range mmap)$*   
 and  $a: a \in range mmap$  and  $S_n: S \ll \{a\}$

```

then obtain n where n: a = mmap n
  by auto
define N where N ≡ LEAST n. S ⊆ mmap ‘ {.. $n$ }
have ∃ n. S ⊆ mmap ‘ {.. $n$ }
  by (metis S finite-nat-iff-bounded finite-subset-image image-mono)
then have S: S ⊆ mmap ‘ {.. $N$ } and minS:  $\bigwedge m. m < N \implies \neg S \subseteq \text{mmap ‘ } \{.. $m$ \}$ 
  unfolding N-def by (meson LeastI-ex not-less-Least)+
have range-mmap-subset: range mmap ⊆ Inf ‘ list.set (F n) ∪ hd (F n) for n
proof (induction n)
  case 0
  then show ?case
    using Eps-subset-hd image-mmap-subset-hd-F by fastforce
next
  case (Suc n)
  then show ?case
    by clarsimp (metis Inf-hd-in-Eps hd-F-in-F image-iff leI mmap-def)
qed
have subM: (Inf ‘ list.set (F N) ∪ hd (F N)) ⊆ M
  by (meson F PowD hd-in-set subsetD sup.boundedI)
have strongly-accepts  $\mathcal{F}$  (insert a S) (Inf ‘ list.set (F N) ∪ hd (F N))
proof (rule insert-closed [unfolded insert-closed-def, rule-format])
  have mmap ‘ {.. $N$ } ⊆ Inf ‘ list.set (F N)
    using Suc-leI hd-F-in-F by (fastforce simp: mmap-def le-eq-less-or-eq)
  with S show Ssub: S ⊆ Inf ‘ list.set (F N)
    by auto
  have S ⊆ mmap ‘ {.. $n$ }
    using Sn S ⟨strict-mono mmap⟩ strict-mono-less
    by (fastforce simp: less-sets-def n image-iff subset-iff Bex-def)
  with leI minS have n ≥ N by blast
  then show a ∈ Eps (Φ (F N))
    using image-mmap-subset-hd-F n by fastforce
  show strongly-accepts  $\mathcal{F}$  S (Inf ‘ list.set (F N) ∪ hd (F N))
  proof (rule ccontr)
    assume ¬ strongly-accepts  $\mathcal{F}$  S (Inf ‘ list.set (F N) ∪ hd (F N))
    then have rejects  $\mathcal{F}$  S (Inf ‘ list.set (F N) ∪ hd (F N))
      using dsM subM unfolding decides-subsets-def
      by (meson F Ssub ⟨finite S⟩ decides-def decides-subset subset-trans)
    moreover have accepts  $\mathcal{F}$  S (range mmap)
      using ⟨inj mmap⟩ acc range-inj-infinite strongly-accepts-imp-accepts by
blast
  ultimately show False
    by (meson range-mmap-subset rejects-subset)
  qed
qed (auto simp: Sn)
then show strongly-accepts  $\mathcal{F}$  (insert a S) (range mmap)
  using range-mmap-subset strongly-accepts-subset by auto
qed
qed

```

## 2.3 Main Theorem

**lemma** *Nash-Williams-1: Ramsey  $\mathcal{F}$  1*

**by** (*auto simp: Ramsey-eq*)

**theorem** *Nash-Williams-2:*

**assumes** *thin-set  $\mathcal{F}$  shows Ramsey  $\mathcal{F}$  2*

**unfolding** *Ramsey-eq*

**proof** *clarify*

**fix**  $f :: \text{nat set} \Rightarrow \text{nat}$  **and**  $M :: \text{nat set}$

**assume** *infinite  $M$  and  $f2: f \in \mathcal{F} \rightarrow \{..<2\}$*

**let**  $?F = \lambda i. f - \{i\} \cap \mathcal{F}$  — *needed with Ramsey-eq, not with Ramsey-def*

**have**  $\mathcal{F}: ?F\ 0 \cup ?F\ 1 = \mathcal{F}$

**using**  *$f2$  less-2-cases by (auto simp: PiE)*

**have**  $\text{fin}\mathcal{F}: \bigwedge X. X \in \mathcal{F} \Longrightarrow \text{finite } X$  **and** *thin:  $\bigwedge i. \text{thin-set } (?F\ i)$*

**using** *assms thin-set-def by auto*

**then obtain**  $N$  **where**  $N \subseteq M$  *infinite  $N$  and  $N: \text{decides-subsets } (?F\ 0)\ N$*

**using**  *$\langle \text{infinite } M \rangle \text{ ex-infinite-decides-subsets by blast}$*

**then consider** *rejects  $(?F\ 0)\ \{\} N$  | strongly-accepts  $(?F\ 0)\ \{\} N$*

**unfolding** *decides-def decides-subsets-def by blast*

**then show**  $\exists N\ i. N \subseteq M \wedge \text{infinite } N \wedge i < 2 \wedge \mathcal{F} \cap \text{Pow } N \subseteq f - \{i\}$

**proof** *cases*

**case** 1

**then have**  $(?F\ 0 \cup ?F\ 1) \cap \text{Pow } N \subseteq f - \{1\}$

**using**  $f2$  *fin $\mathcal{F}$  by (auto simp: Fpow-Pow-finite)*

**then show** *?thesis*

**by** (*metis  $\mathcal{F}$  Suc-1  $\langle N \subseteq M \rangle \langle \text{infinite } N \rangle \text{lessI}$* )

**next**

**case** 2

**then have**  $\S: \bigwedge P. \llbracket P \subseteq N; \bigwedge S. \llbracket S \subseteq P; \text{finite } S \rrbracket \Longrightarrow S \notin ?F\ 0 \rrbracket \Longrightarrow \text{finite } P$

**by** (*auto simp: Fpow-def disjoint-iff*)

**obtain**  $P$  **where**  $P \subseteq N$  *infinite  $P$  and  $P:$*

$\bigwedge S\ n. \llbracket S \subseteq P; \text{finite } S; \text{strongly-accepts } (?F\ 0)\ S\ P; n \in P; S \ll \{n\} \rrbracket$

$\Longrightarrow \text{strongly-accepts } (?F\ 0)\ (\text{insert } n\ S)\ P$

**using** *strongly-accepts-1-19-plus [OF thin  $\langle \text{infinite } N \rangle N$ ] by blast*

**have**  $\mathcal{F} \cap \text{Pow } P \subseteq f - \{0\}$

**proof** (*clarsimp simp: subset-vimage-iff*)

**fix**  $T$

**assume**  $T: T \in \mathcal{F}$  **and**  $T \subseteq P$

**then have** *finite  $T$*

**using** *fin $\mathcal{F}$  by blast*

**moreover have** *strongly-accepts  $(?F\ 0)\ S\ P$  if finite  $S$   $S \subseteq P$  for  $S$*

**using** *that*

**proof** (*induction card  $S$  arbitrary:  $S$* )

**case** (*Suc  $n$* )

**then have** *Seq:  $S = \text{insert } (\text{Sup } S)\ (S - \{\text{Sup } S\})$*

**using** *Sup-nat-def Max-eq-iff by fastforce*

**then have** *sacc: strongly-accepts  $(?F\ 0)\ (S - \{\text{Sup } S\})\ P$*

**by** (*metis Suc card-Diff-singleton diff-Suc-1 finite-Diff insertCI insert-subset*)



```

have  $S - \{Sup\ S\} \ll \{Sup\ S\}$ 
using Suc by (simp add: Sup-nat-def dual-order.strict-iff-order less-sets-def)
then have strongly-accepts ( $?F\ 0$ ) (insert (Sup S) ( $S - \{Sup\ S\}$ )) P
  by (metis P Seq Suc.premis finite-Diff insert-subset sacc)
then show ?case
  using Seq by auto
qed (use 2  $\langle P \subseteq N \rangle$  in auto)
ultimately have  $\exists x \in comparables\ T\ P.\ f\ x = 0 \wedge x \in \mathcal{F}$ 
  using  $\langle T \subseteq P \rangle$   $\langle infinite\ P \rangle$  rejects-def strongly-accepts-def by fastforce
then show  $f\ T = 0$ 
  using T assms thin-set-def comparables-def by force
qed
then show ?thesis
  by (meson  $\langle N \subseteq M \rangle$   $\langle P \subseteq N \rangle$   $\langle infinite\ P \rangle$  less-2-cases-iff subset-trans)
qed
qed

```

**theorem** *Nash-Williams:*

**assumes**  $\mathcal{F}$ : *thin-set*  $\mathcal{F}\ r > 0$  **shows** *Ramsey*  $\mathcal{F}\ r$

**using**  $\langle r > 0 \rangle$

**proof** (*induction r*)

**case** (*Suc r*)

**show** *?case*

**proof** (*cases r < 2*)

**case** *True*

**with** *less-2-cases Nash-Williams-1 Nash-Williams-2* *assms* **show** *?thesis*

**by** (*auto simp: numeral-2-eq-2*)

**next**

**case** *False*

**with** *Suc.IH* **have** *Ram*: *Ramsey*  $\mathcal{F}\ r\ r \geq 2$

**by** *auto*

**show** *?thesis*

**unfolding** *Ramsey-eq*

**proof** *clarify*

**fix** *f* **and** *M* :: *nat set*

**assume** *fim*:  $f \in \mathcal{F} \rightarrow \{.. < Suc\ r\}$

**and** *infinite M*

**let** *?within* =  $\lambda g\ i\ N.\ \mathcal{F} \cap Pow\ N \subseteq g - \{i\}$

**define** *g* **where**  $g \equiv \lambda x.\ if\ f\ x = r\ then\ r-1\ else\ f\ x$

**have** *gim*:  $g \in \mathcal{F} \rightarrow \{.. < r\}$

**using** *fim False* **by** (*force simp: g-def*)

**then obtain** *N i* **where**  $N \subseteq M$  *infinite N*  $i < r$  **and** *i*: *?within g i N*

**using** *Ram*  $\langle infinite\ M \rangle$  **by** (*metis Ramsey-eq*)

**show**  $\exists N\ j.\ N \subseteq M \wedge infinite\ N \wedge j < Suc\ r \wedge ?within\ f\ j\ N$

**proof** (*cases i < r-1*)

**case** *True*

**then have** *?within f i N*

**using**  $\langle N \subseteq M \rangle$   $\langle infinite\ N \rangle$   $\langle i < r \rangle$  *i* **by** (*fastforce simp add: g-def*)

```

then show ?thesis
  by (meson ⟨ $N \subseteq M$ ⟩ ⟨ $i < r$ ⟩ ⟨infinite  $N$ ⟩ less-Suc-eq)
next
case False
then have  $i = r - 1$ 
  using ⟨ $i < r$ ⟩ by linarith
then have null:  $\mathcal{F} \cap \text{Pow } N \subseteq f - \{i, r\}$ 
  using  $i < r$ 
  by (auto simp: g-def split: if-split-asm)
define  $h$  where  $h \equiv \lambda x. \text{if } f x = r \text{ then } 0 \text{ else } f x$ 
have  $h \text{im}: h \in \mathcal{F} \rightarrow \{.. < r\}$ 
  using  $\text{fin } i$  False ⟨ $i < r$ ⟩ by (force simp: h-def)
then obtain  $P j$  where  $P \subseteq N$  infinite  $P$   $j < r$  and  $j: ?\text{within } h j P$ 
  using Ram ⟨ $i < r$ ⟩ ⟨infinite  $N$ ⟩ unfolding Ramsey-eq by metis
have  $\exists i < \text{Suc } r. ?\text{within } f i P$ 
proof (cases  $j=0$ )
  case True
    then have  $\mathcal{F} \cap \text{Pow } P \subseteq f - \{r\}$ 
      using Ram(2) ⟨ $P \subseteq N$ ⟩ ⟨ $i = r - 1$ ⟩  $i j$ 
      unfolding subset-vimage-iff g-def h-def
      by (metis Int-iff Pow-iff Suc-1 diff-is-0-eq insert-iff not-less-eq-eq sub-
set-trans)
    then show ?thesis
      by blast
  next
    case False
      then show ?thesis
        using  $j < r$  by (fastforce simp add: h-def less-Suc-eq)
      qed
    then show ?thesis
      by (meson ⟨ $N \subseteq M$ ⟩ ⟨ $P \subseteq N$ ⟩ ⟨infinite  $P$ ⟩ subset-trans)
    qed
  qed
qed
qed auto
end

```

### 3 Acknowledgements

The author was supported by the ERC Advanced Grant ALEXANDRIA (Project 742178) funded by the European Research Council. Todorčević provided help with the proofs by email.

## References

- [1] C. S. J. A. Nash-Williams. On well-quasi-ordering transfinite sequences. *Mathematical Proceedings of the Cambridge Philosophical Society*, 61(1):33–39, 1965.
- [2] S. Todorćević. *Introduction to Ramsey Spaces*. Princeton University Press, 2010.