

Cardinality of Number Partitions

Lukas Bulwahn

April 19, 2020

Abstract

This entry provides a basic library for number partitions, defines the two-argument partition function through its recurrence relation and relates this partition function to the cardinality of number partitions. The main proof shows that the recursively-defined partition function with arguments n and k equals the cardinality of number partitions of n with exactly k parts. The combinatorial proof follows the proof sketch of Theorem 2.4.1 in Mazur’s textbook “Combinatorics: A Guided Tour” [2]. This entry can serve as starting point for various more intrinsic properties about number partitions, the partition function and related recurrence relations.

Contents

1	Additions to Isabelle’s Main Theories	2
1.1	Addition to Finite-Set Theory	2
1.2	Addition to Set-Interval Theory	2
1.3	Additions to Multiset Theory	2
2	Number Partitions	3
2.1	Number Partitions as $nat \Rightarrow nat$ Functions	3
2.2	Bounds and Finiteness of Number Partitions	4
2.3	Operations of Number Partitions	6
2.4	Number Partitions as Multisets on Natural Numbers	12
	2.4.1 Relationship to Definition on Functions	12
3	Cardinality of Number Partitions	14
3.1	The Partition Function	14
3.2	Cardinality of Number Partitions	15
3.3	Cardinality of Number Partitions as Multisets of Natural Numbers	19
3.4	Cardinality of Number Partitions with only 1-parts	20

1 Additions to Isabelle's Main Theories

```
theory Additions-to-Main
imports HOL-Library.Multiset
begin
```

1.1 Addition to Finite-Set Theory

```
lemma bound-domain-and-range-impl-finitely-many-functions:
```

```
  finite {f::nat⇒nat. (∀i. f i ≤ n) ∧ (∀i≥m. f i = 0)}
```

```
proof (induct m)
```

```
  case 0
```

```
  have eq: {f. (∀i. f i ≤ n) ∧ (∀i. f i = 0)} = {(λ-. 0)} by auto
```

```
  from this show ?case by auto (subst eq; auto)
```

```
next
```

```
  case (Suc m)
```

```
  let ?S = (λ(y, f). f(m := y)) ‘ ({0..n} × {f. (∀i. f i ≤ n) ∧ (∀i≥m. f i =
```

```
0)})
```

```
  {
```

```
    fix g
```

```
    assume ∀i. g i ≤ n ∧ i ≥ Suc m. g i = 0
```

```
    from this have g ∈ ?S
```

```
    by (auto intro: image-eqI[where x=(g m, g(m:=0))])
```

```
  }
```

```
  from this have {f. (∀i. f i ≤ n) ∧ (∀i≥Suc m. f i = 0)} = ?S by auto
```

```
  from this Suc show ?case by simp
```

```
qed
```

1.2 Addition to Set-Interval Theory

```
lemma sum-atMost-remove-nat:
```

```
  assumes k ≤ (n :: nat)
```

```
  shows (∑ i≤n. f i) = f k + (∑ i∈{..n}-{k}. f i)
```

```
using assms by (auto simp add: sum.remove[where x=k])
```

1.3 Additions to Multiset Theory

```
lemma set-mset-Abs-multiset:
```

```
  assumes f ∈ multiset
```

```
  shows set-mset (Abs-multiset f) = {x. f x > 0}
```

```
using assms unfolding set-mset-def by simp
```

```
lemma sum-mset-sum-count:
```

```
  sum-mset M = (∑ i∈set-mset M. count M i * i)
```

```
proof (induct M)
```

```
  show sum-mset {#} = (∑ i∈set-mset {#}. count {#} i * i) by simp
```

```
next
```

```
  fix M x
```

```
  assume hyp: sum-mset M = (∑ i∈set-mset M. count M i * i)
```

```

show  $sum\text{-}mset (add\text{-}mset\ x\ M) = (\sum_{i \in set\text{-}mset (add\text{-}mset\ x\ M)}. count (add\text{-}mset\ x\ M)\ i * i)$ 
proof (cases  $x \in\# M$ )
  assume  $a: \neg x \in\# M$ 
  from this have  $count\ M\ x = 0$  by (meson count-inI)
  from  $\langle \neg x \in\# M \rangle$  this hyp show ?thesis
    by (auto intro!: sum.cong)
next
  assume  $x \in\# M$ 
  have  $sum\text{-}mset (add\text{-}mset\ x\ M) = (\sum_{i \in set\text{-}mset\ M}. count\ M\ i * i) + x$ 
    using hyp by simp
  also have  $\dots = (\sum_{i \in set\text{-}mset\ M - \{x\}}. count\ M\ i * i) + count\ M\ x * x + x$ 
    using  $\langle x \in\# M \rangle$  by (simp add: sum.remove[of - x])
  also have  $\dots = count (add\text{-}mset\ x\ M)\ x * x + (\sum_{i \in set\text{-}mset (add\text{-}mset\ x\ M) - \{x\}}. count (add\text{-}mset\ x\ M)\ i * i)$ 
    by simp
  also have  $\dots = (\sum_{i \in set\text{-}mset (add\text{-}mset\ x\ M)}. count (add\text{-}mset\ x\ M)\ i * i)$ 
    using  $\langle x \in\# M \rangle$  by (simp add: sum.remove[of - x])
  finally show ?thesis .
qed
qed

```

```

lemma sum-mset-eq-sum-on-supersets:
  assumes finite A set-mset M  $\subseteq A$ 
  shows  $(\sum_{i \in set\text{-}mset\ M}. count\ M\ i * i) = (\sum_{i \in A}. count\ M\ i * i)$ 
proof -
  note  $\langle finite\ A \rangle \langle set\text{-}mset\ M \subseteq A \rangle$ 
  moreover have  $\forall i \in A - set\text{-}mset\ M. count\ M\ i * i = 0$ 
    using count-inI by fastforce
  ultimately show ?thesis
    by (auto intro: sum.mono-neutral-cong-left)
qed
end

```

2 Number Partitions

```

theory Number-Partition
imports Additions-to-Main
begin

```

2.1 Number Partitions as $nat \Rightarrow nat$ Functions

```

definition partitions ::  $(nat \Rightarrow nat) \Rightarrow nat \Rightarrow bool$  (infix partitions 50)
where
   $p\ partitions\ n = ((\forall i. p\ i \neq 0 \longrightarrow 1 \leq i \wedge i \leq n) \wedge (\sum_{i \leq n}. p\ i * i) = n)$ 

```

```

lemma partitionsI:
  assumes  $\bigwedge i. p\ i \neq 0 \implies 1 \leq i \wedge i \leq n$ 

```

assumes $(\sum_{i \leq n}. p \ i * i) = n$
shows p partitions n
using *assms unfolding partitions-def by auto*

lemma *partitionsE*:
assumes p partitions n
obtains $\bigwedge i. p \ i \neq 0 \implies 1 \leq i \wedge i \leq n \ (\sum_{i \leq n}. p \ i * i) = n$
using *assms unfolding partitions-def by auto*

lemma *partitions-zero*:
 p partitions $0 \iff p = (\lambda i. 0)$
unfolding *partitions-def by auto*

lemma *partitions-one*:
 p partitions $(\text{Suc } 0) \iff p = (\lambda i. 0)(1 := 1)$
unfolding *partitions-def*
by (*auto split: if-split-asm*) (*auto simp add: fun-eq-iff*)

2.2 Bounds and Finiteness of Number Partitions

lemma *partitions-imp-finite-elements*:
assumes p partitions n
shows *finite* $\{i. 0 < p \ i\}$
proof –
from *assms have* $\{i. 0 < p \ i\} \subseteq \{..n\}$ **by** (*auto elim: partitionsE*)
from this show *?thesis*
using *rev-finite-subset by blast*
qed

lemma *partitions-imp-multiset*:
assumes p partitions n
shows $p \in$ *multiset*
using *assms partitions-imp-finite-elements multiset-def by auto*

lemma *partitions-bounds*:
assumes p partitions n
shows $p \ i \leq n$
proof –
from *assms have* *index-bounds*: $(\forall i. p \ i \neq 0 \implies 1 \leq i \wedge i \leq n)$
and *sum*: $(\sum_{i \leq n}. p \ i * i) = n$
unfolding *partitions-def by auto*
show *?thesis*
proof (*cases* $1 \leq i \wedge i \leq n$)
case *True*
from *True have* $\{..n\} = \text{insert } i \ \{i'. i' \leq n \wedge i' \neq i\}$ **by** *blast*
from *sum[unfolded this] have* $p \ i * i + (\sum_{i \in \{i'. i' \leq n \wedge i' \neq i\}}. p \ i * i) =$
 n **by** *auto*
from this have $p \ i * i \leq n$ **by** *linarith*
from this True show *?thesis using dual-order.trans by fastforce*

```

next
  case False
  from this index-bounds show ?thesis by fastforce
qed
qed

```

```

lemma partitions-parts-bounded:
  assumes p partitions n
  shows  $\text{sum } p \{..n\} \leq n$ 
proof -
  {
    fix i
    assume  $i \leq n$ 
    from assms have  $p\ i \leq p\ i * i$ 
      by (auto elim!: partitionsE)
  }
  from this have  $\text{sum } p \{..n\} \leq (\sum_{i \leq n}. p\ i * i)$ 
    by (auto intro: sum-mono)
  also from assms have  $n: (\sum_{i \leq n}. p\ i * i) = n$ 
    by (auto elim!: partitionsE)
  finally show ?thesis .
qed

```

```

lemma finite-partitions:
  finite {p. p partitions n}
proof -
  have {p. p partitions n}  $\subseteq \{f. (\forall i. f\ i \leq n) \wedge (\forall i. n + 1 \leq i \longrightarrow f\ i = 0)\}$ 
    by (auto elim: partitions-bounds) (auto simp add: partitions-def)
  from this bound-domain-and-range-impl-finitely-many-functions[of n n + 1] show
?thesis
    by (simp add: finite-subset)
qed

```

```

lemma finite-partitions-k-parts:
  finite {p. p partitions n  $\wedge$   $\text{sum } p \{..n\} = k$ }
by (simp add: finite-partitions)

```

```

lemma partitions-remaining-Max-part:
  assumes p partitions n
  assumes  $0 < p\ k$ 
  shows  $\forall i. n - k < i \wedge i \neq k \longrightarrow p\ i = 0$ 
proof (clarify)
  fix i
  assume  $n - k < i \wedge i \neq k$ 
  show  $p\ i = 0$ 
  proof (cases i  $\leq$  n)
    assume  $i \leq n$ 
    from assms have  $n: (\sum_{i \leq n}. p\ i * i) = n$  and  $k \leq n$ 
      by (auto elim: partitionsE)

```

```

have ( $\sum_{i \leq n}. p \ i * i$ ) =  $p \ k * k + (\sum_{i \in \{..n\} - \{k\}}. p \ i * i)$ 
  using  $\langle k \leq n \rangle$  sum-atMost-remove-nat by blast
also have ... =  $p \ i * i + p \ k * k + (\sum_{i \in \{..n\} - \{i, k\}}. p \ i * i)$ 
  using  $\langle i \leq n \rangle \langle i \neq k \rangle$ 
  by (auto simp add: sum.remove[where x=i]) (metis Diff-insert)
finally have eq: ( $\sum_{i \leq n}. p \ i * i$ ) =  $p \ i * i + p \ k * k + (\sum_{i \in \{..n\} - \{i, k\}}. p \ i * i)$  .
show  $p \ i = 0$ 
proof (rule ccontr)
  assume  $p \ i \neq 0$ 
  have upper-bound:  $p \ i * i + p \ k * k \leq n$ 
    using eq n by auto
  have lower-bound:  $p \ i * i + p \ k * k > n$ 
    using  $\langle n - k < i \rangle \langle 0 < p \ k \rangle \langle k \leq n \rangle \langle p \ i \neq 0 \rangle$  mult-eq-if-not-less by auto
  from upper-bound lower-bound show False by simp
qed
next
assume  $\neg (i \leq n)$ 
from this show  $p \ i = 0$ 
  using assms(1) by (auto elim: partitionsE)
qed
qed

```

2.3 Operations of Number Partitions

```

lemma partitions-remove1-bounds:
  assumes partitions:  $p \ partitions \ n$ 
  assumes gr0:  $0 < p \ k$ 
  assumes neq:  $(p(k := p \ k - 1)) \ i \neq 0$ 
  shows  $1 \leq i \wedge i \leq n - k$ 
proof
  from partitions neq show  $1 \leq i$ 
    by (auto elim!: partitionsE split: if-split-asm)
next
from partitions gr0 have n: ( $\sum_{i \leq n}. p \ i * i$ ) =  $n$  and  $k \leq n$ 
  by (auto elim: partitionsE)
show  $i \leq n - k$ 
proof cases
  assume  $k \leq n - k$ 
  from  $\langle k \leq n - k \rangle$  neq show ?thesis
    using partitions-remaining-Max-part[OF partitions gr0] not-le by force
next
  assume  $\neg k \leq n - k$ 
  from this have  $2 * k > n$  by auto
  have  $p \ k = 1$ 
  proof (rule ccontr)
    assume  $p \ k \neq 1$ 
    with gr0 have  $p \ k \geq 2$  by auto
    from this have  $p \ k * k \geq 2 * k$  by simp

```

```

    with  $\langle 2 * k > n \rangle$  have  $p\ k * k > n$  by linarith
    from  $\langle k \leq n \rangle$  this have  $(\sum_{i \leq n}. p\ i * i) > n$ 
      by (simp add: sum-atMost-remove-nat[of k])
    from this  $n$  show False by auto
  qed
  from neq this show ?thesis
    using partitions-remaining-Max-part[OF partitions gr0] leI
    by (auto split: if-split-asm) force
  qed
qed

lemma partitions-remove1:
  assumes partitions:  $p$  partitions  $n$ 
  assumes gr0:  $0 < p\ k$ 
  shows  $p(k := p\ k - 1)$  partitions  $(n - k)$ 
proof (rule partitionsI)
  fix  $i$ 
  assume  $(p(k := p\ k - 1))\ i \neq 0$ 
  from this show  $1 \leq i \wedge i \leq n - k$  using partitions-remove1-bounds partitions
gr0 by blast
next
  from partitions gr0 have  $k \leq n$  by (auto elim: partitionsE)
  have  $(\sum_{i \leq n - k}. (p(k := p\ k - 1))\ i * i) = (\sum_{i \leq n}. (p(k := p\ k - 1))\ i * i)$ 
    using partitions-remove1-bounds partitions gr0 by (auto intro!: sum.mono-neutral-left)
  also have  $\dots = (p\ k - 1) * k + (\sum_{i \in \{..n\} - \{k\}}. (p(k := p\ k - 1))\ i * i)$ 
    using  $\langle k \leq n \rangle$  by (simp add: sum-atMost-remove-nat[where k=k])
  also have  $\dots = p\ k * k + (\sum_{i \in \{..n\} - \{k\}}. p\ i * i) - k$ 
    using gr0 by (simp add: diff-mult-distrib)
  also have  $\dots = (\sum_{i \leq n}. p\ i * i) - k$ 
    using  $\langle k \leq n \rangle$  by (simp add: sum-atMost-remove-nat[of k])
  also from partitions have  $\dots = n - k$ 
    by (auto elim: partitionsE)
  finally show  $(\sum_{i \leq n - k}. (p(k := p\ k - 1))\ i * i) = n - k$  .
qed

lemma partitions-insert1:
  assumes  $p$ :  $p$  partitions  $n$ 
  assumes  $k > 0$ 
  shows  $(p(k := p\ k + 1))$  partitions  $(n + k)$ 
proof (rule partitionsI)
  fix  $i$ 
  assume  $(p(k := p\ k + 1))\ i \neq 0$ 
  from  $p$  this  $\langle k > 0 \rangle$  show  $1 \leq i \wedge i \leq n + k$ 
    by (auto elim!: partitionsE)
next
  have  $(\sum_{i \leq n + k}. (p(k := p\ k + 1))\ i * i) = p\ k * k + (\sum_{i \in \{..n + k\} - \{k\}}. p\ i * i) + k$ 
    by (simp add: sum-atMost-remove-nat[of k])
  also have  $\dots = p\ k * k + (\sum_{i \in \{..n\} - \{k\}}. p\ i * i) + k$ 

```

using p **by** (*auto intro!*: *sum.mono-neutral-right elim!*: *partitionsE*)
also have $\dots = (\sum_{i \leq n}. p\ i * i) + k$
using p **by** (*cases* $k \leq n$) (*auto simp add:* *sum-atMost-remove-nat[of k] elim:*
partitionsE)
also have $\dots = n + k$
using p **by** (*auto elim:* *partitionsE*)
finally show $(\sum_{i \leq n + k}. (p(k := p\ k + 1))\ i * i) = n + k$.
qed

lemma *count-remove1*:

assumes p *partitions* n
assumes $0 < p\ k$
shows $(\sum_{i \leq n - k}. (p(k := p\ k - 1))\ i) = (\sum_{i \leq n}. p\ i) - 1$
proof –

have $k \leq n$ **using** *assms* **by** (*auto elim:* *partitionsE*)
have $(\sum_{i \leq n - k}. (p(k := p\ k - 1))\ i) = (\sum_{i \leq n}. (p(k := p\ k - 1))\ i)$
using *partitions-remove1-bounds* *assms* **by** (*auto intro!*: *sum.mono-neutral-left*)
also have $(\sum_{i \leq n}. (p(k := p\ k - 1))\ i) = p\ k + (\sum_{i \in \{..n\} - \{k\}}. p\ i) - 1$
using $\langle 0 < p\ k \rangle \langle k \leq n \rangle$ **by** (*simp add:* *sum-atMost-remove-nat[of k]*)
also have $\dots = (\sum_{i \in \{..n\}}. p\ i) - 1$
using $\langle k \leq n \rangle$ **by** (*simp add:* *sum-atMost-remove-nat[of k]*)
finally show *?thesis* .
qed

lemma *count-insert1*:

assumes p *partitions* n
shows $sum\ (p(k := p\ k + 1))\ \{..n + k\} = (\sum_{i \leq n}. p\ i) + 1$
proof –
have $(\sum_{i \leq n + k}. (p(k := p\ k + 1))\ i) = p\ k + (\sum_{i \in \{..n + k\} - \{k\}}. p\ i)$
 $+ 1$
by (*simp add:* *sum-atMost-remove-nat[of k]*)
also have $\dots = p\ k + (\sum_{i \in \{..n\} - \{k\}}. p\ i) + 1$
using *assms* **by** (*auto intro!*: *sum.mono-neutral-right elim!*: *partitionsE*)
also have $\dots = (\sum_{i \leq n}. p\ i) + 1$
using *assms* **by** (*cases* $k \leq n$) (*auto simp add:* *sum-atMost-remove-nat[of k]*
elim: *partitionsE*)
finally show *?thesis* .
qed

lemma *partitions-decrease1*:

assumes p : p *partitions* m
assumes *sum:* $sum\ p\ \{..m\} = k$
assumes $p\ 1 = 0$
shows $(\lambda i. p\ (i + 1))$ *partitions* $m - k$
proof –
from p **have** $p\ 0 = 0$ **by** (*auto elim!*: *partitionsE*)
{
fix i
assume *neg:* $p\ (i + 1) \neq 0$


```

from  $p$  this  $\langle p\ 1 = 0 \rangle$  have  $1 \leq i$ 
  by (fastforce elim!: partitionsE simp add: le-Suc-eq)
moreover have  $i \leq m - k$ 
proof (rule ccontr)
  assume i-greater:  $\neg i \leq m - k$ 
  from  $p$  have  $s: (\sum_{i \leq m}. p\ i * i) = m$ 
    by (auto elim!: partitionsE)
  from  $p$  sum have  $k \leq m$ 
    using partitions-parts-bounded by fastforce
  from neq p have  $i + 1 \leq m$  by (auto elim!: partitionsE)
  from i-greater have  $i > m - k$  by simp
  have ineq1:  $i + 1 > (m - k) + 1$ 
    using i-greater by simp
  have ineq21:  $(\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j * j) \geq (\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j)$ 
    using  $\langle p\ 0 = 0 \rangle$  not-less by (fastforce intro!: sum-mono)
  have ineq22a:  $(\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j) = (\sum_{j \leq m}. p\ j) - 1$ 
    using  $\langle i + 1 \leq m \rangle$  neq by (simp add: sum.remove[where x=i + 1])
  have ineq22:  $(\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j) \geq k - 1$ 
    using sum neq ineq22a by auto
  have ineq2:  $(\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j * j) \geq k - 1$ 
    using ineq21 ineq22 by auto
  have  $(\sum_{i \leq m}. p\ i * i) = p(i + 1) * (i + 1) + (\sum_{i \in \{..m\} - \{i + 1\}}. p\ i * i)$ 
    using  $\langle i + 1 \leq m \rangle$  neq
    by (subst sum.remove[where x=i + 1]) auto
  also have  $\dots = (i + 1) + (\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j * j)$ 
    using  $\langle i + 1 \leq m \rangle$  neq
    by (subst sum.remove[where x=i + 1 and g=λj. (p(i + 1 := p(i + 1) - 1))\ j * j])
    (auto simp add: mult-eq-iff)
  finally have  $(\sum_{i \leq m}. p\ i * i) = i + 1 + (\sum_{j \leq m}. (p(i + 1 := p(i + 1) - 1))\ j * j)$ 
    moreover have  $\dots > m$  using ineq1 ineq2  $\langle k \leq m \rangle$   $\langle p(i + 1) \neq 0 \rangle$  by
linarith
  ultimately have  $(\sum_{i \leq m}. p\ i * i) > m$  by simp
  from  $s$  this show False by simp
qed
ultimately have  $1 \leq i \wedge i \leq m - k$  ..
} note bounds = this
show  $(\lambda i. p(i + 1))$  partitions  $m - k$ 
proof (rule partitionsI)
  fix  $i$ 
  assume  $p(i + 1) \neq 0$ 
  from bounds this show  $1 \leq i \wedge i \leq m - k$  .
next
  have geq:  $\forall i. p\ i * i \geq p\ i$ 
    using  $\langle p\ 0 = 0 \rangle$  not-less by fastforce
  have  $(\sum_{i \leq m - k}. p(i + 1) * i) = (\sum_{i \leq m}. p(i + 1) * i)$ 

```

using *bounds* by (auto intro: sum.mono-neutral-left)
 also have ... = $(\sum_{i \in \text{Suc } \{..m\}} p\ i * (i - 1))$
 by (auto simp add: sum.reindex)
 also have ... = $(\sum_{i \leq \text{Suc } m} p\ i * (i - 1))$
 using $\langle p\ 0 = 0 \rangle$ by (simp add: atMost-Suc-eq-insert-0 zero-notin-Suc-image)
 also have ... = $(\sum_{i \leq m} p\ i * (i - 1))$
 using *p* by (auto elim!: partitionsE)
 also have ... = $(\sum_{i \leq m} p\ i * i - p\ i)$
 by (simp add: diff-mult-distrib2)
 also have ... = $(\sum_{i \leq m} p\ i * i) - (\sum_{i \leq m} p\ i)$
 using *geq* by (simp only: sum-subtractf-nat)
 also have ... = $m - k$ using *sum p* by (auto elim!: partitionsE)
 finally show $(\sum_{i \leq m - k} p\ (i + 1) * i) = m - k$.
 qed
 qed

lemma *partitions-increase1*:

assumes *partitions*: *p partitions m - k*
 assumes *k*: *sum p {..m - k} = k*
 shows $(\lambda i. p\ (i - 1))$ *partitions m*
 proof (rule *partitionsI*)
 fix *i*
 assume $p\ (i - 1) \neq 0$
 from *partitions this k* show $1 \leq i \wedge i \leq m$
 by (cases *k*) (auto elim!: partitionsE)
 next
 from *k partitions* have $k \leq m$
 using *linear partitions-zero* by force
 have *eq-0*: $\forall i > m - k. p\ i = 0$ using *partitions* by (auto elim!: partitionsE)
 from *partitions* have *s*: $(\sum_{i \leq m - k} p\ i * i) = m - k$ by (auto elim!: partitionsE)
 have $(\sum_{i \leq m} p\ (i - 1) * i) = (\sum_{i \leq \text{Suc } m} p\ (i - 1) * i)$
 using *partitions k* by (cases *k*) (auto elim!: partitionsE)
 also have $(\sum_{i \leq \text{Suc } m} p\ (i - 1) * i) = (\sum_{i \leq m} p\ i * (i + 1))$
 by (subst sum.atMost-Suc-shift) simp
 also have ... = $(\sum_{i \leq m - k} p\ i * (i + 1))$
 using *eq-0* by (auto intro: sum.mono-neutral-right)
 also have ... = $(\sum_{i \leq m - k} p\ i * i) + (\sum_{i \leq m - k} p\ i)$ by (simp add: sum.distrib)
 also have ... = $m - k + k$ using *s k* by simp
 also have ... = m using $\langle k \leq m \rangle$ by simp
 finally show $(\sum_{i \leq m} p\ (i - 1) * i) = m$.
 qed

lemma *count-decrease1*:

assumes *p*: *p partitions m*
 assumes *sum*: *sum p {..m} = k*
 assumes *p 1 = 0*
 shows *sum* $(\lambda i. p\ (i + 1)) \{..m - k\} = k$

proof –
from p **have** $p\ 0 = 0$ **by** (*auto elim!:* *partitionsE*)
have $\text{sum } (\lambda i. p\ (i + 1)) \{..m - k\} = \text{sum } (\lambda i. p\ (i + 1)) \{..m\}$
using *partitions-decrease1* [*OF assms*]
by (*auto intro:* *sum.mono-neutral-left elim!:* *partitionsE*)
also have $\dots = \text{sum } (\lambda i. p\ (i + 1)) \{0..m\}$ **by** (*simp add:* *atLeast0AtMost*)
also have $\dots = \text{sum } (\lambda i. p\ i) \{Suc\ 0.. Suc\ m\}$
by (*simp only:* *One-nat-def add-Suc-right add-0-right sum.shift-bounds-cl-Suc-ivl*)
also have $\dots = \text{sum } (\lambda i. p\ i) \{.. Suc\ m\}$
using $\langle p\ 0 = 0 \rangle$ **by** (*simp add:* *atLeast0AtMost sum-shift-lb-Suc0-0*)
also have $\dots = \text{sum } (\lambda i. p\ i) \{.. m\}$
using p **by** (*auto elim!:* *partitionsE*)
also have $\dots = k$
using *sum* **by** *simp*
finally show *?thesis* .
qed

lemma *count-increase1*:

assumes *partitions:* p *partitions* $m - k$
assumes k : $\text{sum } p \{..m - k\} = k$
shows $(\sum_{i \leq m} p\ (i - 1)) = k$

proof –

have $p\ 0 = 0$ **using** *partitions* **by** (*auto elim!:* *partitionsE*)
have $(\sum_{i \leq m} p\ (i - 1)) = (\sum_{i \in \{1..m\}} p\ (i - 1))$
using $\langle p\ 0 = 0 \rangle$ **by** (*auto intro:* *sum.mono-neutral-cong-right*)
also have $(\sum_{i \in \{1..m\}} p\ (i - 1)) = (\sum_{i \leq m - 1} p\ i)$
proof (*cases m*)

case 0

from this show *?thesis* **using** $\langle p\ 0 = 0 \rangle$ **by** *simp*

next

case (*Suc m'*)

{
fix x **assume** $Suc\ 0 \leq x\ x \leq m$
from this Suc have $x \in Suc\ ' \{..m'\}$
by (*auto intro!:* *image-eqI* [**where** $x=x - 1$])
}

from this Suc show *?thesis*

by (*intro sum.reindex-cong* [*of Suc*]) *auto*

qed

also have $(\sum_{i \leq m - 1} p\ i) = (\sum_{i \leq m} p\ i)$

proof –

{
fix i
assume $0 < p\ i\ i \leq m$
from assms this have $i \leq m - 1$
using $\langle p\ 0 = 0 \rangle$ *partitions-increase1* **by** (*cases k*) (*auto elim!:* *partitionsE*)
}

from this show *?thesis*

by (*auto intro:* *sum.mono-neutral-cong-left*)

qed
also have ... = $(\sum_{i \leq m - k}. p\ i)$
using *partitions* **by** (*auto intro: sum.mono-neutral-right elim!: partitionsE*)
also have ... = k **using** k **by** *auto*
finally show *?thesis* .
qed

2.4 Number Partitions as Multisets on Natural Numbers

definition *number-partition* :: $nat \Rightarrow nat\ multiset \Rightarrow bool$
where

number-partition $n\ N = (sum\ mset\ N = n \wedge 0 \notin\# N)$

2.4.1 Relationship to Definition on Functions

lemma *count-partitions-iff*:

count N *partitions* $n \iff number\ partition\ n\ N$

proof

assume *count* N *partitions* n

from *this* **have** $(\forall i. count\ N\ i \neq 0 \longrightarrow 1 \leq i \wedge i \leq n)$ $(\sum_{i \leq n}. count\ N\ i * i)$
= n

unfolding *Number-Partition.partitions-def* **by** *auto*

moreover from *this* **have** $set\ mset\ N \subseteq \{..n\}$ **by** *auto*

moreover have *finite* $\{..n\}$ **by** *auto*

ultimately have $sum\ mset\ N = n$

using *sum-mset-sum-count sum-mset-eq-sum-on-supersets* **by** *presburger*

moreover have $0 \notin\# N$

using $(\forall i. count\ N\ i \neq 0 \longrightarrow 1 \leq i \wedge i \leq n)$ **by** *auto*

ultimately show *number-partition* $n\ N$

unfolding *number-partition-def* **by** *auto*

next

assume *number-partition* $n\ N$

from *this* **have** $sum\ mset\ N = n$ **and** $0 \notin\# N$

unfolding *number-partition-def* **by** *auto*

{

fix i

assume *count* $N\ i \neq 0$

have $1 \leq i \wedge i \leq n$

proof

from $(0 \notin\# N)$ $(count\ N\ i \neq 0)$ **show** $1 \leq i$

using *Suc-le-eq* **by** *auto*

from $(sum\ mset\ N = n)$ $(count\ N\ i \neq 0)$ **show** $i \leq n$

using *multi-member-split* **by** *fastforce*

qed

}

moreover from $(sum\ mset\ N = n)$ **have** $(\sum_{i \leq n}. count\ N\ i * i) = n$

by (*metis atMost-iff calculation finite-atMost not-in-iff subsetI sum-mset-eq-sum-on-supersets sum-mset-sum-count*)

ultimately show *count* N *partitions* n

by (*rule partitionsI*) *auto*

qed

lemma *partitions-iff-Abs-multiset*:

p partitions $n \iff \text{finite } \{x. 0 < p x\} \wedge \text{number-partition } n \text{ (Abs-multiset } p)$

proof

assume p partitions n

from *this* have *bounds*: $(\forall i. p i \neq 0 \implies 1 \leq i \wedge i \leq n)$

and *sum*: $(\sum_{i \leq n}. p i * i) = n$

unfolding *partitions-def* **by** *auto*

from $\langle p \text{ partitions } n \rangle$ have $p \in \text{multiset}$ **by** (*rule partitions-imp-multiset*)

from $\langle p \text{ partitions } n \rangle$ have *finite* $\{x. 0 < p x\}$

by (*rule partitions-imp-finite-elements*)

moreover from $\langle p \in \text{multiset} \rangle$ *bounds* have $\neg 0 \in \# \text{Abs-multiset } p$

using *count-eq-zero-iff* **by** *force*

moreover from $\langle p \in \text{multiset} \rangle$ *this sum* have *sum-mset* $(\text{Abs-multiset } p) = n$

proof –

have $(\sum_{i \in \{x. 0 < p x\}}. p i * i) = (\sum_{i \leq n}. p i * i)$

using *bounds* **by** (*auto intro: sum.mono-neutral-cong-left*)

from $\langle p \in \text{multiset} \rangle$ *this sum* **show** *sum-mset* $(\text{Abs-multiset } p) = n$

by (*simp add: sum-mset-sum-count set-mset-Abs-multiset*)

qed

ultimately show *finite* $\{x. 0 < p x\} \wedge \text{number-partition } n \text{ (Abs-multiset } p)$

unfolding *number-partition-def* **by** *auto*

next

assume *finite* $\{x. 0 < p x\} \wedge \text{number-partition } n \text{ (Abs-multiset } p)$

from *this* have *finite* $\{x. 0 < p x\}$ $0 \notin \# \text{Abs-multiset } p$ *sum-mset* $(\text{Abs-multiset } p) = n$

unfolding *number-partition-def* **by** *auto*

from $\langle \text{finite } \{x. 0 < p x\} \rangle$ have $p \in \text{multiset}$ **by** (*simp add: multiset-def*)

from $\langle p \in \text{multiset} \rangle$ have $(\sum_{i \in \{x. 0 < p x\}}. p i * i) = n$

using $\langle \text{sum-mset } (\text{Abs-multiset } p) = n \rangle$

by (*simp add: sum-mset-sum-count set-mset-Abs-multiset*)

have *bounds*: $\bigwedge i. p i \neq 0 \implies 1 \leq i \wedge i \leq n$

proof

fix i

assume $p i \neq 0$

from $\langle \neg 0 \in \# \text{Abs-multiset } p \rangle$ $\langle p \in \text{multiset} \rangle$ have $p 0 = 0$

using *count-inI* **by** *force*

from *this* $\langle p i \neq 0 \rangle$ **show** $1 \leq i$

by (*metis One-nat-def leI less-Suc0*)

show $i \leq n$

proof (*rule ccontr*)

assume $\neg i \leq n$

from *this* have $i > n$

using *le-less-linear* **by** *blast*

from *this* $\langle p i \neq 0 \rangle$ have $p i * i > n$

by (*auto simp add: less-le-trans*)

from $\langle p i \neq 0 \rangle$ have $(\sum_{i \in \{x. 0 < p x\}}. p i * i) = p i * i + (\sum_{i \in \{x. 0 < p x\} - \{i\}}. p i * i)$

```

    using ⟨finite {x. 0 < p x}⟩
    by (subst sum.insert-remove[symmetric]) (auto simp add: insert-absorb)
  also from ⟨p i * i > n⟩ have ... > n by auto
  finally show False using ⟨(∑ i∈{x. 0 < p x}. p i * i) = n⟩ by blast
qed
qed
moreover have (∑ i≤n. p i * i) = n
proof -
  have (∑ i≤n. p i * i) = (∑ i∈{x. 0 < p x}. p i * i)
    using bounds by (auto intro: sum.mono-neutral-cong-right)
  from this show ?thesis
    using ⟨(∑ i∈{x. 0 < p x}. p i * i) = n⟩ by simp
qed
ultimately show p partitions n by (auto intro: partitionsI)
qed

```

```

lemma size-nat-multiset-eq:
  fixes N :: nat multiset
  assumes number-partition n N
  shows size N = sum (count N) {..n}
proof -
  have set-mset N ⊆ {..sum-mset N}
    by (auto dest: multi-member-split)
  have size N = sum (count N) (set-mset N)
    by (rule size-multiset-overloaded-eq)
  also have ... = sum (count N) {..sum-mset N}
    using ⟨set-mset N ⊆ {..sum-mset N}⟩
    by (auto intro: sum.mono-neutral-cong-left count-inI)
  finally show ?thesis
    using ⟨number-partition n N⟩
    unfolding number-partition-def by auto
qed

```

end

3 Cardinality of Number Partitions

```

theory Card-Number-Partitions
imports Number-Partition
begin

```

3.1 The Partition Function

```

fun Partition :: nat ⇒ nat ⇒ nat
where
  Partition 0 0 = 1
| Partition 0 (Suc k) = 0
| Partition (Suc m) 0 = 0

```

| $Partition (Suc m) (Suc k) = Partition m k + Partition (m - k) (Suc k)$

lemma *Partition-less*:

assumes $m < k$

shows $Partition m k = 0$

using *assms* **by** (induct m k rule: *Partition.induct*) auto

lemma *Partition-sum-Partition-diff*:

assumes $k \leq m$

shows $Partition m k = (\sum_{i \leq k} Partition (m - k) i)$

using *assms* **by** (induct m k rule: *Partition.induct*) auto

lemma *Partition-parts1*:

$Partition (Suc m) (Suc 0) = 1$

by (induct m) auto

lemma *Partition-diag*:

$Partition (Suc m) (Suc m) = 1$

by (induct m) auto

lemma *Partition-diag1*:

$Partition (Suc (Suc m)) (Suc m) = 1$

by (induct m) auto

lemma *Partition-parts2*:

shows $Partition m 2 = m \text{ div } 2$

proof (induct m rule: *nat-less-induct*)

fix m

assume *hypothesis*: $\forall n < m. Partition n 2 = n \text{ div } 2$

have $(m = 0 \vee m = 1) \vee m \geq 2$ **by** auto

from *this* **show** $Partition m 2 = m \text{ div } 2$

proof

assume $m = 0 \vee m = 1$

from *this* **show** *?thesis* **by** (auto simp add: numerals(2))

next

assume $2 \leq m$

from *this* **obtain** m' **where** m': $m = Suc (Suc m')$ **by** (*metis add-2-eq-Suc le-Suc-ex*)

from *hypothesis* *this* **have** $Partition m' 2 = m' \text{ div } 2$ **by** *simp*

from *this* m' **show** *?thesis*

using *Partition-parts1* *Partition.simps(4)*[of *Suc m' Suc 0*] *div2-Suc-Suc*

by (*simp* add: numerals(2) del: *Partition.simps*)

qed

qed

3.2 Cardinality of Number Partitions

lemma *set-rewrite1*:

$\{p. p \text{ partitions } Suc m \wedge \text{sum } p \{..Suc m\} = Suc k \wedge p 1 \neq 0\}$

= $(\lambda p. p(1 := p 1 + 1)) \text{ ‘ } \{p. p \text{ partitions } m \wedge \text{sum } p \{..m\} = k\} \text{ (is } ?S = ?T)$

proof

```

{
  fix p
  assume assms: p partitions Suc m sum p {..Suc m} = Suc k 0 < p 1
  have p(1 := p 1 - 1) partitions m
    using assms by (metis partitions-remove1 diff-Suc-1)
  moreover have  $(\sum_{i \leq m. (p(1 := p 1 - 1)) i} = k$ 
    using assms by (metis count-remove1 diff-Suc-1)
  ultimately have p(1 := p 1 - 1)  $\in \{p. p \text{ partitions } m \wedge \text{sum } p \{..m\} = k\}$ 
by simp
  moreover have p = p(1 := p 1 - 1, 1 := (p(1 := p 1 - 1)) 1 + 1)
    using <0 < p 1> by auto
  ultimately have p  $\in (\lambda p. p(1 := p 1 + 1)) \text{ ‘ } \{p. p \text{ partitions } m \wedge \text{sum } p \{..m\} = k\}$  by blast
}
from this show ?S  $\subseteq$  ?T by blast
next
{
  fix p
  assume assms: p partitions m sum p {..m} = k
  have (p(1 := p 1 + 1)) partitions Suc m (is ?g1)
    using assms by (metis partitions-insert1 Suc-eq-plus1 zero-less-one)
  moreover have sum (p(1 := p 1 + 1)) {..Suc m} = Suc k (is ?g2)
    using assms by (metis count-insert1 Suc-eq-plus1)
  moreover have (p(1 := p 1 + 1)) 1  $\neq$  0 (is ?g3) by auto
  ultimately have ?g1  $\wedge$  ?g2  $\wedge$  ?g3 by simp
}
from this show ?T  $\subseteq$  ?S by auto
qed

```

lemma set-rewrite2:

$\{p. p \text{ partitions } m \wedge \text{sum } p \{..m\} = k \wedge p 1 = 0\}$
= $(\lambda p. (\lambda i. p (i - 1))) \text{ ‘ } \{p. p \text{ partitions } (m - k) \wedge \text{sum } p \{..m - k\} = k\}$
(is ?S = ?T)

proof

```

{
  fix p
  assume assms: p partitions m sum p {..m} = k p 1 = 0
  have  $(\lambda i. p (i + 1))$  partitions m - k
    using assms partitions-decrease1 by blast
  moreover from assms have sum  $(\lambda i. p (i + 1)) \{..m - k\} = k$ 
    using assms count-decrease1 by blast
  ultimately have  $(\lambda i. p (i + 1)) \in \{p. p \text{ partitions } m - k \wedge \text{sum } p \{..m - k\} = k\}$  by simp
  moreover have p =  $(\lambda i. p ((i - 1) + 1))$ 
proof (rule ext)
  fix i show p i = p (i - 1 + 1)

```



```

    using assms by (cases i) (auto elim!: partitionsE)
  qed
  ultimately have  $p \in (\lambda p. (\lambda i. p (i - 1))) \text{ ' } \{p. p \text{ partitions } m - k \wedge \text{sum } p \{..m - k\} = k\}$  by auto
}
from this show  $?S \subseteq ?T$  by auto
next
{
  fix p
  assume assms:  $p \text{ partitions } m - k \text{ sum } p \{..m - k\} = k$ 
  from assms have  $(\lambda i. p (i - 1)) \text{ partitions } m$  (is ?g1)
    using partitions-increase1 by blast
  moreover from assms have  $(\sum_{i \leq m}. p (i - 1)) = k$  (is ?g2)
    using count-increase1 by blast
  moreover from assms have  $p 0 = 0$  (is ?g3)
    by (auto elim!: partitionsE)
  ultimately have  $?g1 \wedge ?g2 \wedge ?g3$  by simp
}
from this show  $?T \subseteq ?S$  by auto
qed

```

theorem *card-partitions-k-parts*:

```

  card  $\{p. p \text{ partitions } n \wedge (\sum_{i \leq n}. p i) = k\} = \text{Partition } n k$ 
proof (induct n k rule: Partition.induct)
  case 1
  have eq:  $\{p. p = (\lambda x. 0) \wedge p 0 = 0\} = \{(\lambda x. 0)\}$  by auto
  show card  $\{p. p \text{ partitions } 0 \wedge \text{sum } p \{..0\} = 0\} = \text{Partition } 0 0$ 
    by (simp add: partitions-zero eq)
  next
  case (2 k)
  have eq:  $\{p. p = (\lambda x. 0) \wedge p 0 = \text{Suc } k\} = \{\}$  by auto
  show card  $\{p. p \text{ partitions } 0 \wedge \text{sum } p \{..0\} = \text{Suc } k\} = \text{Partition } 0 (\text{Suc } k)$ 
    by (simp add: partitions-zero eq)
  next
  case (3 m)
  have eq:  $\{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = 0\} = \{\}$ 
    by (fastforce elim!: partitionsE simp add: le-Suc-eq)
  from this show card  $\{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = 0\} = \text{Partition } (\text{Suc } m) 0$ 
    by (simp only: Partition.simps card-empty)
  next
  case (4 m k)
  let ?set1 =  $\{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = \text{Suc } k \wedge p 1 \neq 0\}$ 
  let ?set2 =  $\{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = \text{Suc } k \wedge p 1 = 0\}$ 
  have finite  $\{p. p \text{ partitions } \text{Suc } m\}$ 
    by (simp add: finite-partitions)
  from this have finite-sets: finite ?set1 finite ?set2 by simp+
  have set-eq:  $\{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = \text{Suc } k\} = ?set1 \cup ?set2$  by auto

```

have *disjoint*: $?set1 \cap ?set2 = \{\}$ **by** *auto*
have *inj1*: *inj-on* $(\lambda p. p(1 := p 1 + 1)) \{p. p \text{ partitions } m \wedge \text{sum } p \{..m\} = k\}$
by (*auto intro!*: *inj-onI*) (*metis diff-Suc-1 fun-upd-idem-iff fun-upd-upd*)
have *inj2*: *inj-on* $(\lambda p i. p(i - 1)) \{p. p \text{ partitions } m - k \wedge \text{sum } p \{..m - k\} = \text{Suc } k\}$
by (*auto intro!*: *inj-onI simp add: fun-eq-iff*) (*metis add-diff-cancel-right'*)
have *card1*: $\text{card } ?set1 = \text{Partition } m k$
using *inj1* 4(1) **by** (*simp only: set-rewrite1 card-image*)
have *card2*: $\text{card } ?set2 = \text{Partition } (m - k) (\text{Suc } k)$
using *inj2* 4(2) **by** (*simp only: set-rewrite2 card-image diff-Suc-Suc*)
have $\text{card } \{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = \text{Suc } k\} = \text{Partition } m k$
 $+ \text{Partition } (m - k) (\text{Suc } k)$
using *finite-sets disjoint* **by** (*simp only: set-eq card-Un-disjoint card1 card2*)
from *this* **show** $\text{card } \{p. p \text{ partitions } \text{Suc } m \wedge \text{sum } p \{..\text{Suc } m\} = \text{Suc } k\} =$
 $\text{Partition } (\text{Suc } m) (\text{Suc } k)$
by *auto*
qed

theorem *card-partitions*:

$\text{card } \{p. p \text{ partitions } n\} = (\sum k \leq n. \text{Partition } n k)$

proof –

have *seteq*: $\{p. p \text{ partitions } n\} = \bigcup ((\lambda k. \{p. p \text{ partitions } n \wedge (\sum i \leq n. p i) = k\}) ' \{..n\})$

by (*auto intro: partitions-parts-bounded*)

have *finite*: $\bigwedge k. \text{finite } \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\}$

by (*simp add: finite-partitions*)

have $\text{card } \{p. p \text{ partitions } n\} = \text{card } (\bigcup ((\lambda k. \{p. p \text{ partitions } n \wedge (\sum i \leq n. p i) = k\}) ' \{..n\}))$

using *finite* **by** (*simp add: seteq*)

also **have** $\dots = (\sum x \leq n. \text{card } \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = x\})$

using *finite* **by** (*subst card-UN-disjoint*) *auto*

also **have** $\dots = (\sum k \leq n. \text{Partition } n k)$

by (*simp add: card-partitions-k-parts*)

finally **show** *?thesis* .

qed

lemma *card-partitions-atmost-k-parts*:

$\text{card } \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\} = \text{Partition } (n + k) k$

proof –

have $\text{card } \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\} =$

$\text{card } (\bigcup ((\lambda k'. \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k'\}) ' \{..k\}))$

proof –

have $\{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\} =$

$(\bigcup k' \leq k. \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k'\})$ **by** *auto*

from *this* **show** *?thesis* **by** *simp*

qed

also **have** $\text{card } (\bigcup ((\lambda k'. \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k'\}) ' \{..k\})) =$

$\text{sum } (\lambda k'. \text{card } \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k'\}) \{..k\}$

using *finite-partitions-k-parts* **by** (*subst card-UN-disjoint*) *auto*

also have $\dots = \text{sum } (\lambda k'. \text{Partition } n k') \{..k\}$
using *card-partitions-k-parts* **by** *simp*
also have $\dots = \text{Partition } (n + k) k$
using *Partition-sum-Partition-diff* **by** *simp*
finally show *?thesis* .
qed

3.3 Cardinality of Number Partitions as Multisets of Natural Numbers

lemma *bij-betw-multiset-number-partition-with-size*:

bij-betw count $\{N. \text{number-partition } n N \wedge \text{size } N = k\} \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\}$

proof (*rule* *bij-betw-byWitness*[**where** $f' = \text{Abs-multiset}$])

show $\forall N \in \{N. \text{number-partition } n N \wedge \text{size } N = k\}. \text{Abs-multiset } (\text{count } N) = N$

using *count-inverse* **by** *blast*

show $\forall p \in \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\}. \text{count } (\text{Abs-multiset } p) = p$

by (*auto simp add: multiset-def partitions-imp-finite-elements*)

show *count* ' $\{N. \text{number-partition } n N \wedge \text{size } N = k\} \subseteq \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\}$

by (*auto simp add: count-partitions-iff size-nat-multiset-eq*)

show *Abs-multiset* ' $\{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\} \subseteq \{N. \text{number-partition } n N \wedge \text{size } N = k\}$

using *partitions-iff-Abs-multiset size-nat-multiset-eq partitions-imp-multiset* **by** *fastforce*

qed

lemma *bij-betw-multiset-number-partition-with-atmost-size*:

bij-betw count $\{N. \text{number-partition } n N \wedge \text{size } N \leq k\} \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\}$

proof (*rule* *bij-betw-byWitness*[**where** $f' = \text{Abs-multiset}$])

show $\forall N \in \{N. \text{number-partition } n N \wedge \text{size } N \leq k\}. \text{Abs-multiset } (\text{count } N) = N$

using *count-inverse* **by** *blast*

show $\forall p \in \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\}. \text{count } (\text{Abs-multiset } p) = p$

by (*auto simp add: multiset-def partitions-imp-finite-elements*)

show *count* ' $\{N. \text{number-partition } n N \wedge \text{size } N \leq k\} \subseteq \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\}$

by (*auto simp add: count-partitions-iff size-nat-multiset-eq*)

show *Abs-multiset* ' $\{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq k\} \subseteq \{N. \text{number-partition } n N \wedge \text{size } N \leq k\}$

using *partitions-iff-Abs-multiset size-nat-multiset-eq partitions-imp-multiset* **by** *fastforce*

qed

theorem *card-number-partitions-with-atmost-k-parts*:

shows *card* $\{N. \text{number-partition } n N \wedge \text{size } N \leq x\} = \text{Partition } (n + x) x$

proof –

have *bij-betw count* $\{N. \text{number-partition } n N \wedge \text{size } N \leq x\} \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq x\}$
by (*rule bij-betw-multiset-number-partition-with-atmost-size*)
from this have *card* $\{N. \text{number-partition } n N \wedge \text{size } N \leq x\} = \text{card} \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq x\}$
by (*rule bij-betw-same-card*)
also have *card* $\{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} \leq x\} = \text{Partition } (n + x) x$
by (*rule card-partitions-atmost-k-parts*)
finally show *?thesis* .
qed

theorem *card-partitions-with-k-parts*:

card $\{N. \text{number-partition } n N \wedge \text{size } N = k\} = \text{Partition } n k$

proof –

have *bij-betw count* $\{N. \text{number-partition } n N \wedge \text{size } N = k\} \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\}$
by (*rule bij-betw-multiset-number-partition-with-size*)
from this have *card* $\{N. \text{number-partition } n N \wedge \text{size } N = k\} = \text{card} \{p. p \text{ partitions } n \wedge \text{sum } p \{..n\} = k\}$
by (*rule bij-betw-same-card*)
also have $\dots = \text{Partition } n k$ **by** (*rule card-partitions-k-parts*)
finally show *?thesis* .
qed

3.4 Cardinality of Number Partitions with only 1-parts

lemma *number-partition1-eq-replicate-mset*:

$\{N. (\forall n. n \in \# N \longrightarrow n = 1) \wedge \text{number-partition } n N\} = \{\text{replicate-mset } n 1\}$

proof

show $\{N. (\forall n. n \in \# N \longrightarrow n = 1) \wedge \text{number-partition } n N\} \subseteq \{\text{replicate-mset } n 1\}$

proof

fix N

assume $N: N \in \{N. (\forall n. n \in \# N \longrightarrow n = 1) \wedge \text{number-partition } n N\}$

have $N = \text{replicate-mset } n 1$

proof (*rule multiset-eqI*)

fix i

have *count* $N 1 = \text{sum-mset } N$

proof *cases*

assume $N = \{\#\}$

from this show *?thesis* **by** *auto*

next

assume $N \neq \{\#\}$

from this N **have** $1 \in \# N$ **by** *blast*

from this N **show** *?thesis*

by (*auto simp add: sum-mset-sum-count sum.remove[where x=1] simp del: One-nat-def*)

qed

from N **this show** *count* $N i = \text{count } (\text{replicate-mset } n 1) i$

```

      unfolding number-partition-def by (auto intro: count-inI)
    qed
    from this show  $N \in \{\text{replicate-mset } n \ 1\}$  by simp
  qed
next
  show  $\{\text{replicate-mset } n \ 1\} \subseteq \{N. (\forall n. n \in\# N \longrightarrow n = 1) \wedge \text{number-partition } n \ N\}$ 
  unfolding number-partition-def by auto
qed

lemma card-number-partitions-with-only-parts-1-eq-1:
  assumes  $n \leq x$ 
  shows  $\text{card } \{N. (\forall n. n \in\# N \longrightarrow n = 1) \wedge \text{number-partition } n \ N \wedge \text{size } N \leq x\} = 1$  (is card ?N = -)
proof -
  have  $\forall N \in \{N. (\forall n. n \in\# N \longrightarrow n = 1) \wedge \text{number-partition } n \ N\}. \text{size } N = n$ 
  unfolding number-partition1-eq-replicate-mset by simp
  from this number-partition1-eq-replicate-mset  $\langle n \leq x \rangle$  have  $?N = \{\text{replicate-mset } n \ 1\}$  by auto
  from this show ?thesis by simp
qed

lemma card-number-partitions-with-only-parts-1-eq-0:
  assumes  $x < n$ 
  shows  $\text{card } \{N. (\forall n. n \in\# N \longrightarrow n = 1) \wedge \text{number-partition } n \ N \wedge \text{size } N \leq x\} = 0$  (is card ?N = -)
proof -
  have  $\forall N \in \{N. (\forall n. n \in\# N \longrightarrow n = 1) \wedge \text{number-partition } n \ N\}. \text{size } N = n$ 
  unfolding number-partition1-eq-replicate-mset by simp
  from this number-partition1-eq-replicate-mset  $\langle x < n \rangle$  have  $?N = \{\}$  by auto
  from this show ?thesis by (simp only: card-empty)
qed

end

```

References

- [1] M. Junker. Diskrete Algebraische Strukturen, 2010. German lecture notes from Mathematisches Institut Albert-Ludwigs-Universität Freiburg.
- [2] D. R. Mazur. *Combinatorics: a guided tour*. MAA textbooks. Mathematical Association of America, 2010.