Spivey’s Generalized Recurrence for Bell Numbers

Lukas Bulwahn

August 16, 2018

Abstract

This entry defines the Bell numbers [1] as the cardinality of set partitions for a carrier set of given size, and derives Spivey’s generalized recurrence relation for Bell numbers [2] following his elegant and intuitive combinatorial proof.

As the set construction for the combinatorial proof requires construction of three intermediate structures, the main difficulty of the formalization is handling the overall combinatorial argument in a structured way. The introduced proof structure allows us to compose the combinatorial argument from its subparts, and supports to keep track how the detailed proof steps are related to the overall argument. To obtain this structure, this entry uses set monad notation for the set construction’s definition, introduces suitable predicates and rules, and follows a repeating structure in its Isar proof.

Contents

1 Bell Numbers and Spivey’s Generalized Recurrence 1
  1.1 Preliminaries . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
    1.1.1 Additions to FuncSet . . . . . . . . . . . . . . . . . . . . 2
    1.1.2 Additions for Injectivity Proofs . . . . . . . . . . . . . . 2
  1.2 Definition of Bell Numbers . . . . . . . . . . . . . . . . . . 3
  1.3 Construction of the Partitions . . . . . . . . . . . . . . . . . . 4
  1.4 Injectivity of the Set Construction . . . . . . . . . . . . . . . 8
  1.5 The Generalized Bell Recurrence Relation . . . . . . . . . . . 9
  1.6 Corollaries of the Generalized Bell Recurrence . . . . . . . . 12

1 Bell Numbers and Spivey’s Generalized Recurrence

theory Bell-Numbers
imports
  HOL-Library.FuncSet
  HOL-Library.Monad-Syntax
1.1 Preliminaries

1.1.1 Additions to FuncSet

**lemma** extensional-funcset-ext:
assumes \( f \in A \to E \) \( B \) \( g \in A \to E \) \( B \)
assumes \( \forall x. \ x \in A \implies f x = g x \)
shows \( f = g \)
using assms by (metis PiE-iff extensionalityI)

1.1.2 Additions for Injectivity Proofs

**lemma** inj-on-impl-inj-on-image:
assumes \( \text{inj-on } f A \)
assumes \( \forall x. \ x \in X \implies x \subseteq A \)
shows \( \text{inj-on } (('f) X) \)
using assms by (meson inj-onI inj-on-image-eq-iff)

**lemma** injectivity-union:
assumes \( A \cup B = C \cup D \)
assumes \( P \ A \ P \ C \)
assumes \( Q \ B \ Q \ D \)
\( \forall S \ T. \ P \ S \implies Q \ T \implies S \cap T = \{} \)
shows \( A = C \land B = D \)
using assms Int-Un-distrib Int-commute inf-sup-absorb by blast+

**lemma** injectivity-image:
assumes \( f ' A = g ' A \)
assumes \( \forall x \in A. \ \text{invert } (f x) = x \land \text{invert } (g x) = x \)
shows \( \forall x \in A. \ f x = g x \)
using assms by (metis (no-types, lifting) image-iff)

**lemma** injectivity-image-union:
assumes \( (\lambda X. \ X \cup F X) \ ' P = (\lambda X. \ X \cup G X) \ ' P' \)
assumes \( \forall X \in P. \ X \subseteq A \land X \in P'. \ X \subseteq A \)
assumes \( \forall y \in F X. \ y \notin A \land X \in P'. \ \forall y \in G X. \ y \notin A \)
shows \( P = P' \)
proof
show \( P \subseteq P' \)
proof
fix \( X \)
assume \( X \in P \)
from assms(1) this obtain \( X' \) where \( X' \in P' \) and \( X \cup F X = X' \cup G X' \)
by (metis imageE image-eqI)
moreover from assms(2,4) \( :X \in P' \) have \( X: (X \cup F X) \cap A = X \) by auto
moreover from \( \text{assms}(3,5) \) \( X' \in P \) have \( X' \cap (X' \cup G X') \cap A = X' \) by auto ultimately have \( X = X' \) by simp from this \( X' \in P \) show \( X \in P' \) by auto qed

next show \( P' \subseteq P \)
proof
fix \( X' \)
assume \( X' \in P' \)
from \( \text{assms}(1) \) this obtain \( X \) where \( X \in P \) and \( X \cup F X = X' \cup G X' \)
by (metis \( \text{imageE \ image-eqI} \))
moreover from \( \text{assms}(2,4) \) \( X \in P \) have \( X : (X \cup F X) \cap A = X \) by auto
moreover from \( \text{assms}(3,5) \) \( X' \in P' \) have \( X' : (X' \cup G X') \cap A = X' \) by auto ultimately have \( X = X' \) by simp from this \( X \in P \) show \( X' \in P \) by auto qed

qed

1.2 Definition of Bell Numbers

definition\(\text{Bell} :: \mathbb{N} \Rightarrow \mathbb{N}\)
where
\(\text{Bell} \; n = \text{card} \{ P. \text{partition-on} \{0..<n\} \; P \}\)

lemma\(\text{Bell-altdef} : \)
assumes \(\text{finite } A\)
shows \(\text{Bell} \; (\text{card} \; A) = \text{card} \{ P. \text{partition-on} \; A \; P \}\)
proof –
from \(\text{finite } A\) obtain \( f \) where \( \text{bij} : \text{bij-betw} \; \{0..<\text{card} \; A\} \; A \)
using \(\text{ex-bij-betw-nat-finite} \) by blast
from this have \( \text{inj} : \text{inj-on} \; f \; \{0..<\text{card} \; A\} \)
using \(\text{bij-betw-imp-inj-on} \) by blast
from \(\text{bij} \) have \( \text{image-f-eq} : A = f \; \{0..<\text{card} \; A\} \)
using \(\text{bij-betw-imp-surj-on} \) by blast
have \( \forall x \in \{ P. \text{partition-on} \; \{0..<\text{card} \; A\} \; P \}. x \subseteq \text{Pow} \; \{0..<\text{card} \; A\} \)
by (auto elim: \(\text{partition-onE} \))
from this \(\text{inj} \) have \( \text{inj-on} \; (('\) \; ((') \; f)) \; \{ P. \text{partition-on} \; \{0..<\text{card} \; A\} \; P \} \)
by (intro \(\text{inj-on-impl-inj-on-image[of - Pow \; \{0..<\text{card} \; A\}] \}
\text{inj-on-impl-inj-on-image[of - \{0..<\text{card} \; A\}]} \) blast+
moreover from \(\text{inj} \) have \( \text{bij-betw} \; ((') \; ((') \; f)) \; \{ P. \text{partition-on} \; \{0..<\text{card} \; A\} \; P \} = \{ P. \text{partition-on} \; A \; P \} \)
by (subst \(\text{image-f-eq, auto elim!: \text{set-of-partition-on-map} }\))
ultimately have \( \text{bij-betw} \; ((') \; ((') \; f)) \; \{ P. \text{partition-on} \; \{0..<\text{card} \; A\} \; P \} \; \{ P. \text{partition-on} \; A \; P \} \)
by (auto intro: \(\text{bij-betw-imageI} \))
from this \(\text{finite } A\) show \( ?\text{thesis} \)
unfolding \(\text{Bell-def} \)
by (subst bij-betw-iff-card[symmetric]) (auto intro: finitely-many-partition-on)
qed

lemma Bell-0:
    Bell 0 = 1
by (auto simp add: Bell-def partition-on-empty)

1.3 Construction of the Partitions

definition construct-partition-on :: 'a set ⇒ 'a set ⇒ 'a set set set
where
    construct-partition-on B C =
     do
        k ← {0..card B};
        j ← {0..card C};
        P ← {P. partition-on C P ∧ card P = j};
        B' ← {B'. B' ⊆ B ∧ card B' = k};
        Q ← {Q. partition-on B' Q};
        f ← (B - B') → E P;
        P' ← {λ X. X ∪ {x ∈ B - B'. f x = X} ' P};
        {P' ∪ Q}
    )

lemma construct-partition-on:
    assumes finite B finite C
    assumes B ∩ C = {}
    shows construct-partition-on B C = {P. partition-on (B ∪ C) P}
proof (rule set-eqI)
    fix Q'
    assume Q' ∈ construct-partition-on B C
    from this obtain j k P P' Q B' f
        where j ≤ card C
        and k ≤ card B
        and P: partition-on C P ∧ card P = j
        and B': B' ⊆ B ∧ card B' = k
        and Q: partition-on B' Q
        and f: f ∈ B - B' → E P
        and P': P' = (λX. X ∪ {x ∈ B - B'. f x = X}) ' P
        and Q': Q' = P' ∪ Q
    unfolding construct-partition-on-def by auto
    from P f have partition-on (B - B' ∪ C) P'
        unfolding P' using (B ∩ C = {});
        by (intro partition-on-insert-elements) auto
    from this Q have partition-on ((B - B' ∪ C) ∪ B') Q'
        unfolding Q' using B' (∧ B ∩ C = {}); by (auto intro: partition-on-union)
    from this have partition-on (B ∪ C) Q'
        using B' by (metis Diff-partition sup.assoc sup.commute)
    from this show Q' ∈ {P. partition-on (B ∪ C) P} by auto
next
fix $Q'$

assume $Q': Q' \in \{ Q', \text{partition-on (} B \cup C \text{)} \}$

from $Q'$ have $\{ \} \notin Q'$ by (auto elim!: partition-onE)

obtain $Q$ where $Q: Q = ((\lambda X. \text{if } X \subseteq B \text{ then } X \text{ else } \{ \}) \cdot Q') - \{ \} \text{ by blast}$

obtain $P'$ where $P': P' = ((\lambda X. \text{if } X \subseteq B \text{ then } \{ \text{ else } X \}) \cdot Q') - \{ \} \text{ by blast}$

from $P' Q \{ \} \notin Q'$ have $Q'$-prop: $Q' = P' \cup Q$ by auto

have $P'$-nosubset: $\forall X \in P'. \neg X \subseteq B$

unfolding $P'$ by auto

moreover have $\forall X \in P'. X \subseteq B \cup C$

using $Q' P'$ by (auto elim: partition-onE)

ultimately have $P'$-witness: $\forall X \in P'. \exists x. x \in X \cap C$

using $(B \cap C = \{ \})$ by fastforce

obtain $B'$ where $B': B' = \bigcup Q$ by blast

have $Q$-prop: partition-on $B'$ $Q$

using $B' Q$ $Q'$-prop partition-on-split2 mem-Collect-eq by blast

have $\bigcup P' = B - B' \cup C$

proof

have $\bigcup Q' = B \cup C \forall X \in Q', \forall X' \in Q', X \neq X' \rightarrow X \cap X' = \{ \}$

using $Q'$ unfolding partition-on-def disjoint-def by auto

from this show $\bigcup P' \subseteq B - B' \cup C$

unfolding $P' B' Q$ by auto blast

next

show $B - B' \cup C \subseteq \bigcup P'$

proof

fix $x$

assume $x \in B - B' \cup C$

from this obtain $X$ where $X: x \in X X \in Q'$

using $Q'$ by (metis Diff_iff Un_iff mem-Collect-eq partition-on-partition-on-unique)

have $\forall X \in Q', X \subseteq B \rightarrow X \subseteq B'$

unfolding $B' Q$ by auto

from this $X (x \in B - B' \cup C)$ have $\neg X \subseteq B$

using $(B \cap C = \{ \})$ by auto

from this $X \in Q'$ have $X \in P'$ using $P'$ by auto

from this $x \in X$ show $x \in \bigcup P'$ by auto

qed

qed

from this have partition-on-P': partition-on (B - B' \cup C) P'

using partition-on-split1 $Q'$-prop $Q'$ mem-Collect-eq by fastforce

obtain $P$ where $P: P = (\lambda X. X \cap C) \cdot P'$ by blast

from $P$ partition-on-P' $P'$-witness have partition-on $C P$

using partition-on-intersect-on-elements by auto

obtain $f$ where $f: f = (\lambda x. \text{if } x \in B - B' \text{ then } (THE X. x \in X \land X \in P') \cap C \text{ else undefined})$ by blast

have $P'$-prop: $P' = (\lambda X. X \cup \{ x \in B - B', f x = X \}) \cdot P$

proof

{ fix $X$

assume $X \in P'$


have X-subset: \( X \subseteq (B - B') \cup C \)

using partition-on-P'. \( \forall X \in P' \) by (auto elim: partition-onE)

have \( X = X \cap C \cup \{ x \in B - B'. f x = X \cap C \} \)

proof

{ 
  fix \( x \)
  assume \( x \in X \)
  from this X-subset have \( x \in (B - B') \cup C \) by auto
  from this have \( x \in X \cap C \cup \{ xa \in B - B'. f xa = X \cap C \} \)
  proof
    assume \( x \in C \)
    from this \( \langle x \in X \rangle \) show \( \langle \text{thesis} \rangle \) by simp
  next
    assume \( x \in B - B' \)
    from partition-on-P'. \( \langle x \in X \rangle \) \( \langle X \in P' \rangle \) have (THE \( X, x \in X \land X \in P' \))
      by (simp add: partition-on-the-part-eq)
    from \( \langle x \in B - B' \rangle \) this show \( \langle \text{thesis} \rangle \) unfolding \( f \) by auto
  qed
}

from this show \( X \subseteq X \cap C \cup \{ x \in B - B'. f x = X \cap C \} \) by auto

next

show \( X \cap C \cup \{ xa \in B - B'. f xa = X \cap C \} \subseteq X \)

proof

  fix \( x \)
  assume \( x \in X \cap C \cup \{ xa \in B - B'. f xa = X \cap C \} \)
  from this have \( x \in X \)
  proof
    assume \( x \in X \cap C \)
    from this show \( \langle \text{thesis} \rangle \) by simp
  next
    assume \( x \in B - B' \)
    from x-in X-subset have eq: (THE \( X, x \in X \land X \in P' \)) \( \cap C = X \cap C \)
      unfolding \( f \) by auto
    from P'-nosubset :\( \langle X \in P' \rangle \) have \( \neg X \subseteq B \) by simp
    from this have \( X \cap C \neq \{ \} \)
      using X-subset assms(3) by blast
  from this obtain \( y \) where \( y \in X \cap C \) by auto
  from this eq have y-in: \( y \in (THE X, x \in X \land X \in P') \cap C \) by simp
  from y y-in have \( y \in X \) \( y \in (THE X, x \in X \land X \in P') \) by auto
  moreover from y have \( \exists! X. y \in X \land X \in P' \)
    using partition-on-P' by (simp add: partition-on-partition-on-unique)
  moreover have (THE \( X, x \in X \land X \in P' \)) \( \in P' \)
    using ext1 by (rule the1I2) auto
  ultimately have (THE \( X, x \in X \land X \in P' \)) = \( X \) using \( \langle X \in P' \rangle \) by auto
  from this ext1 show \( \langle \text{thesis} \rangle \) by (auto intro: the1I2)

from this ext1 show \( \langle \text{thesis} \rangle \) by (auto intro: the1I2)
qed
qed

from $\langle X \in P \rangle$ this have $X \in (\forall X. \ X \cup \{x \in B - B'. \ f x = X\}) \ \wedge P$
unfolding $P$ by simp

} from this show $P \subseteq (\forall X. \ X \cup \{x \in B - B'. \ f x = X\}) \ \wedge P$ ..

next

{ fix $x$
assumes x-in-image: $x \in (\forall X. \ X \cup \{x \in B - B'. \ f x = X\}) \ \wedge P$

{ fix $X$
assumes $X \in P$

have $\{x \in B - B'. \ f x = X \cap C\} = \{x \in B - B'. \ x \in X\}$

proof -

{ fix $x$
assumes $x \in B - B'$
from this have $ex1: \exists! X. \ x \in X \wedge X \in P$

using partition-on-P' by (auto intro!: partition-on-partition-on-unique)

from this have in-p: $(\forall X. \ x \in X \wedge X \in P') \ \wedge x-in: \ x \in (\forall X. \ x \in X \wedge X \in P')$
by (metis (mono-tags, lifting) theI)+

have $f x = X \cap C \iff (\forall X. \ x \in X \wedge X \in P') \cap C = X \cap C$
using $\{x \in B - B'\}$ unfolding $f$ by auto

also have ... $\iff (\forall X. \ x \in X \wedge X \in P') = X$

proof
assumes $(\forall X. \ x \in X \wedge X \in P') = X$
from this show $(\forall X. \ x \in X \wedge X \in P') \cap C = X \cap C$ by auto

next
assumes $(\forall X. \ x \in X \wedge X \in P') \cap C = X \cap C$

have $(\forall X. \ x \in X \wedge X \in P') \cap X = \{\}$
using $\{\}$-witness $(\forall X. \ x \in X \wedge X \in P') \cap C = X \cap C \cap X \in P'$ by fastforce

from this show $(\forall X. \ x \in X \wedge X \in P') = X$

using partition-on-P'[unfolded partition-on-def disjoint-def] in-p $\langle X \in P \rangle$ by metis

qed
also have ... $\iff x \in X$

using $ex1 (X \in P)$ x-in by (auto; metis (no-types, lifting) the-equality)
finally have $f x = X \cap C \iff x \in X$ .

} from this show $\forall \ the\ thesis$ by auto

qed

moreover have $X \subseteq B - B' \cup C$

using partition-on-P' $\langle X \in P \rangle$ by (blast elim: partition-onE)
ultimately have $X \cap C \cup \{x \in B. \ x \notin B' \wedge f x = X \cap C\} = X$ by auto

}
from this x-in-image have $x \in P'$ unfolding $P$ by auto

qed

from partition-on-$P'$ have $f$-prop: $f \in (B - B') \rightarrow_E P$
unfolding $f$ $P$ by (auto simp add: partition-on-the-part-mem)

from $Q B'$ have $B' \subseteq B$ by auto
obtain $k$ where $k: k = \text{card } B'$ by blast
from (finite $B$) ($B' \subseteq B$) $k$ have $k$-prop: $k \in \{0 \ldots \text{card } B\}$ by (simp add: card-mono)

obtain $j$ where $j: j = \text{card } P$ by blast
from $j$ (partition-on $C P$) have $j$-prop: $j \in \{0 \ldots \text{card } C\}$
by (simp add: assms(2) partition-on-le-set-elements)

from (partition-on $C P$) $j$ have $P$-prop: partition-on $C P \land \text{card } P = j$ by auto
from $k$ ($B' \subseteq B$) have $B'$-prop: $B' \subseteq B \land \text{card } B' = k$ by auto

show $Q' \in \text{construct-partition-on } B \ C$
using $j$-prop $k$-prop $P$-prop $B'$-prop $Q$-prop $P'$-prop $Q'$-prop
unfolding construct-partition-on-def
by (auto simp del: atLeastAtMost-iff) blast

qed

1.4 Injectivity of the Set Construction

lemma injectivity:
assumes $B \cap C = \{\}$
assumes $P$: (partition-on $C P \land \text{card } P = j$) \land (partition-on $C P' \land \text{card } P' = j'$)
assumes $B'$: ($B' \subseteq B \land \text{card } B' = k$) \land ($B'' \subseteq B \land \text{card } B'' = k'$)
assumes $Q$: partition-on $B'$ $Q$ \land partition-on $B''$ $Q'$
assumes $f$: $f \in B - B' \rightarrow_E P \land g \in B - B'' \rightarrow_E P'$
assumes $P'$: $P'' = (\lambda X. X \cup \{x \in B - B'. f x = X\}) \cdot P \land$
$P''' = (\lambda X. X \cup \{x \in B - B''. g x = X\}) \cdot P'$
assumes eq-result: $P'' \cup Q = P''' \cup Q'$
shows $f = g$ and $Q = Q'$ and $B' = B''$
and $P = P'$ and $j = j'$ and $k = k'$

proof –
have $P$-nonempty-sets: $\forall X \in P, \exists c \in C. c \in X \forall X \in P', \exists c \in C. c \in X$
using $P$ by (force elim: partition-on$E)$

have 1: $\forall X \in P''. \exists c \in C. c \in X \forall X \in P'''. \exists c \in C. c \in X$
using $P'$ $P$-nonempty-sets by fastforce

have 2: $\forall X \in Q. \forall x \in X, x \notin C \forall X \in Q'. \forall x \in X, x \notin C$
using $(B \cap C = \{\}) \cdot Q B'$ by (auto elim: partition-on$E$)

from eq-result have $P'' = P'''$ and $Q = Q'$
by (auto dest: injectivity-union[OF 1 2])

from this $Q$ show $Q = Q'$ and $B' = B''$
by (auto intro!: partition-on-eq-implies-eq-carrier)

have subset-$C$: $\forall X \in P. X \subseteq C \forall X \in P'. X \subseteq C$
using $P$ by (auto elim: partition-on$E$)

have eq-image: $\langle \lambda X. X \cup \{x \in B - B'. f x = X\} \cdot P \rangle = (\lambda X. X \cup \{x \in B -$
B', g x = X\}) \cdot P'
  
  using P' \cdot P'' = P''' \ by \ auto
  
  from \ this \ (B \cap C = \{\}) \ show \ P = P'
  
  by \ (auto \ dest: \ injectivity-image-union[of \ subset-C])

have eq2: \((\lambda X \cdot X \cup \{x \in B - B', f x = X\}) \cdot P = (\lambda X \cdot X \cup \{x \in B - B', g x = X\}) \cdot P\)
  
  using \ (P = P') \cdot B' = B'' \ eq-image \ by \ simp
  
  from \ P \ have \ P-props: \forall X \in P. X \subseteq C \forall X \in P. X \neq \{\} \ by \ (auto \ elim: partition-onE)
  
  have invert: \forall X \in P. (X \cup \{x \in B - B', f x = X\}) \cap C = X \wedge (X \cup \{x \in B - B', g x = X\}) \cap C = X
  
  using \ (B \cap C = \{\}) \ P-props \ by \ auto
  
  have eq3: \forall X \in P. (X \cup \{x \in B - B', f x = X\}) = (X \cup \{x \in B - B', g x = X\})
  
  using \ injectivity-image[of \ eq2 \ invert] \ by \ blast
  
  have eq4: \forall X \in P. \{x \in B - B', f x = X\} = \{x \in B - B', g x = X\}
  
  proof
  
  fix \ X
  
  assume \ X \in P
  
  from \ this \ P \ have \ X \subseteq C \ by \ (auto \ elim: partition-onE)
  
  have disjoint: \ X \cap \{x \in B - B', f x = X\} = \{\} \ X \cap \{x \in B - B', g x = X\} = \{\}
  
  using \ (B \cap C = \{\}) \ \cdot \ X \subseteq C \ by \ auto
  
  from \ eq3 \ X \in P \ have \ X \cup \{x \in B - B', f x = X\} = X \cup \{x \in B - B', g x = X\}
  
  by \ (auto \ intro: injectivity-anion)

qed

from \ eq4 \ f \ have \ eq5: \forall b \in B - B'. f b = g b \ by \ blast

from \ eq5 \ (\ b' = B''' \ \cdot \ P = P'') \ show \ eq6: f = g \ by \ (auto \ intro: extensional-funcset-ext)

from \ P \ \cdot \ P'' \ show \ j = j' \ by \ simp

from \ B' (\ b' = B''' \ \cdot \ show \ k = k' \ by \ simp

qed

1.5 The Generalized Bell Recurrence Relation

**Theorem Bell-eq:**

\[ Bell \ (n + m) = (\sum k \leq n. \sum j \leq m. \ j \cdot (n - k) \cdot Stirling \ m j \cdot (n \ choose \ k) \cdot Bell \ k) \]

**Proof**

- define \( A \) where \( A = \{0..<n + m\} \)
- define \( B \) where \( B = \{0..<n\} \)
- define \( C \) where \( C = \{n..<n + m\} \)
- have \( A = B \cup C \ B \cap C = \{\} \) finite \( B \) card \( B = n \) finite \( C \) card \( C = m \)
- unfolding \( A-def \ B-def \ C-def \ by \ auto \)
- have step1: Bell \ (n + m) = card \ (\ P. \ partition-on A \ P) \)
- unfolding \( Bell-def A-def .. \)
- from \( A = B \cup C \) \( B \cap C = \{\} \) (finite \( B \) \ finite \( C \) \)
- have step2: card \ (\ P. \ partition-on A \ P) = card \ (\ construct-partition-on B \ C) \)
by (simp add: construct-partition-on)

note injectivity = injectivity[OF "B ∩ C = {}"]

let ?expr = do 
  k ← {0..card B};
  j ← {0..card C};
  P ← {P. partition-on C P ∧ card P = j};
  B′ ← {B’. B′ ⊆ B ∧ card B′ = k};
  Q ← {Q. partition-on B′ Q};
  f ← (B − B′) → E P;
  P′ ← {λX. X ∪ {x ∈ B − B′. f x = X}} ' P;
  {P′ ∪ Q}

let ?S ≝ ?comp = ?expr 
{
  fix k
  assume k: k ∈ {..card B}
  let ?expr = ?comp k
  let ?S ≝ ?comp = ?expr
  { 
    fix j
    assume j ∈ {.. card C}
    let ?expr = ?comp j
    let ?S ≝ ?comp = ?expr
    from (finite C) have finite ?S 
      by (intro finite-Collect-conj disjI1 finitely-many-partition-on)
    { 
      fix P
      assume P: P ∈ {P. partition-on C P ∧ card P = j}
      from this have partition-on C P by simp
      let ?expr = ?comp P
      let ?S ≝ ?comp = ?expr
      have finite P
        using P (finite C) by (auto intro: finite-elements)
      from (finite B) have finite ?S by (auto simp add: finite-subset)
    }
    moreover 
    { 
      fix B′
      assume B′: B′ ∈ {B’. B′ ⊆ B ∧ card B′ = k}
      from this have B′ ⊆ B by simp
      let ?expr = ?comp B′
      let ?S ≝ ?comp = ?expr
      from (finite B) have finite B′
        using B′ by (auto simp add: finite-subset)
      from (finite B') have finite {Q. partition-on B' Q}
        by (rule finitely-many-partition-on)
      moreover 
      { 
        fix Q
        assume Q: Q ∈ {Q. partition-on B' Q}
      }
  }
}

10
let ?expr = ?comp Q
let ?S ≫? comp = ?expr
{
  fix f
  assume f ∈ B − B’ →E P
  let ?expr = ?comp f
  let ?S ≫? comp = ?expr
  have disjoint-family-on ?comp ?S
    by (auto intro: disjoint-family-onI)
  from this have card ?expr = 1
    by (simp add: card-bind-constant)
  moreover have finite ?expr
    by (simp add: finite-bind)
  ultimately have finite ?expr ∧ card ?expr = 1 by blast
}
moreover have finite ?S
  using ⟨finite B⟩ ⟨finite P⟩ by (auto intro: finite-PiE)
moreover have disjoint-family-on ?comp ?S
  using P B’ Q
  by (injectivity-solver rule: local.injectivity(1))
moreover have card ?S = j * (n − k)
proof −
  have card (B − B’) = n − k
    using B’ ⟨finite B’⟩ ⟨card B = n⟩
    by (subst card-Diff-subset) auto
  from this show ?thesis
    using ⟨finite ?S⟩ P
    by (subst card-PiE) (simp add: prod-constant)+
qed
ultimately have finite ?expr
  using ⟨finite ?S⟩ ⟨finite {P. partition-on C P ∧ card P = j}⟩
  by (auto intro!: finite-bind)
ultimately have finite ?expr ∧ card ?expr = j * (n − k) by blast
} note inner = this
moreover have card ?S = Bell k
  using B’ ⟨finite B’⟩ by (auto simp add: Bell-altdef[symmetric])
moreover have disjoint-family-on ?comp ?S
  using P B’
  by (injectivity-solver rule: local.injectivity(2))
ultimately have card ?expr = j * (n − k) * Bell k
  by (subst card-bind-constant) auto
moreover have finite ?expr
  using inner ⟨finite ?S⟩ by (auto intro: finite-bind)
ultimately have finite ?expr ∧ card ?expr = j * (n − k) * Bell k by blast
} note inner = this
moreover have card ?S = n choose k

11
using \( \langle \text{card } B = n \rangle \langle \text{finite } B \rangle \) by (simp add: n-subsets)
moreover have disjoint-family-on \(?comp \ ?S\)
using \( P \)
by (injectivity-solver rule: local.injectivity(3))
ultimately have \( \text{card } \?expr = j \cdot (n - k) \ast (n \text{ choose } k) \ast \text{Bell } k \)
by (subst card-bind-constant) auto
moreover have finite \( \?expr \)
using inner \( \langle \text{finite } \?S \rangle \)
by (auto intro: finite-bind)
ultimately have finite \( \?expr \wedge \text{card } \?expr = j \cdot (n - k) \ast (n \text{ choose } k) \ast \text{Bell } k \)
by blast

1.6 Corollaries of the Generalized Bell Recurrence

corollary Bell-Stirling-eq:
\( \text{Bell } m = (\sum_{j \leq m} \text{Stirling } m \ j) \)
proof –
have $\text{Bell } m = \text{Bell } (0 + m)$ by simp
also have ... $= \left( \sum j \leq m. \text{Stirling } m \ j \right)$
unfolding Bell-eq[of 0] by (simp add: Bell-0)
finally show $?\ thesis$ .
qed

corollary Bell-recursive-eq:
$\text{Bell } (n + 1) = \left( \sum k \leq n. \binom{n}{k} \ast \text{Bell } k \right)$
unfolding Bell-eq[of - 1] by simp

end

References
