Concurrent Computation of Binomial Coefficient

Dimitrios Yfantidis (3938)

Aristotle University of Thessaloniki,
Faculty of Sciences, Department of Informatics

January 2024

Concurrent Computation of Binomial Coefficient

Dimitrios Yfantidis (3938)

Aristotle University of Thessaloniki,
Faculty of Sciences, Department of Informatics

January 2024

Abstract

This presentation accounts for an academic assignment on the lesson **Concurrent Programming & Software Safety**.

Abstract

This presentation accounts for an academic assignment on the lesson **Concurrent Programming & Software Safety**.

The assignment in question mainly demands the implementation of a concurrent program, as formulated by Manna & Pnueli, to compute the binomial coefficient:

$$\binom{n}{k} = \frac{n!}{(n-k)! \cdot k!} = \frac{n \cdot (n-2) \cdot \ldots \cdot (n-k+1)}{1 \cdot 2 \cdot \ldots \cdot k}$$

Assignment Prompt

- **Hypothesis:** Suppose that one procedure computes the numerator while another procedure computes the denominator.

Assignment Prompt

- **Hypothesis:** Suppose that one procedure computes the numerator while another procedure computes the denominator.
- **Tip:** Prove that $i!$ is a divisor of $j \cdot (j+1) \cdot \ldots \cdot (j+i-1)$. This way the numerator's procedure can fetch partially computed results from the denominator's procedure and performs the division immediately, so that its partially computed results don't grow too big. E.g: $1 \cdot 2$ divides $10 \cdot 9$, $1 \cdot 2 \cdot 3$ divides $10 \cdot 9 \cdot 8$, etc.

Mathematical Preliminary Work

1. Argument $\mathcal{I}(i, j)$:

$$i! \mid j \cdot (j+1) \cdot \ldots \cdot (j+i-1)$$

2. Suppose $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ and

$$f(m, n) = m \cdot (m+1) \cdot \ldots \cdot (m+n-1) = \prod_{k=0}^{n-1} (m+k)$$

3. Argument $\mathcal{I}(i, j)$ can be written as:

$$i! \mid f(j, i)$$

Mathematical Preliminary Work

Induction by i

- Trivial case, $\mathcal{I}(1, j)$: $1!|f(j, 1) \Leftrightarrow 1|j$ (true $\forall j \in \mathbb{N}$)
- Induction hypothesis: $i! \mid f(j, i)$
- Induction step: $i \rightarrow i + 1$

Mathematical Preliminary Work

Induction by i

- Trivial case, $\mathcal{I}(1, j)$: $1! | f(j, 1) \Leftrightarrow 1 | j$ (true $\forall j \in \mathbb{N}$)
- Induction hypothesis: $i! \mid f(j, i)$
- Induction step: $i \rightarrow i + 1$

Induction by j (within the induction step of i)

- Trivial case, $\mathcal{I}(i + 1, 0)$:

$$f(0, i + 1) = 0 \cdot (0 + 1) \cdot \ldots \cdot (0 + i) = 0, \forall i \in \mathbb{N}$$

  Thus, the argument $\mathcal{I}(i + 1, 0)$ holds true as $(i + 1)! \mid 0, \forall i$
- Induction hypothesis: $(i + 1)! \mid f(j, i + 1)$
- Induction step: $j \rightarrow j + 1$

$$\begin{aligned} f(j + 1, i + 1) &= (j + 1) \cdot \left[(j + 1) + 1\right] \cdot \ldots \cdot \left[(j + 1) + (i + 1) - 1\right] \\ &= (j + 1) \cdot (j + 2) \cdot \ldots \cdot (j + i) \cdot (j + i + 1) \\ &= (i + 1) \cdot (j + 1) \cdot \ldots \cdot (j + i) + j \cdot (j + 1) \cdot \ldots \cdot (j + i) \\ &= (i + 1) \cdot f(j + 1, i) + f(j, i + 1) \end{aligned}$$

Mathematical Preliminary Work

- The first term is divisible by $(i+1)!$ because of the induction hypothesis for i, thus:

$$(i+1)! \mid (i+1) \cdot f(j+1, i)$$

Mathematical Preliminary Work

- The first term is divisible by $(i+1)!$ because of the induction hypothesis for i, thus:

$$(i+1)! \mid (i+1) \cdot f(j+1, i)$$

- The second term is divisible by $(i+1)!$ because of the induction hypothesis for j, thus:

$$(i+1)! \mid f(j, i+1)$$

Mathematical Preliminary Work

- The first term is divisible by $(i+1)!$ because of the induction hypothesis for i, thus:

$$(i+1)! \mid (i+1) \cdot f(j+1, i)$$

- The second term is divisible by $(i+1)!$ because of the induction hypothesis for j, thus:

$$(i+1)! \mid f(j, i+1)$$

- As a consequence:

$$(i+1)! \mid (i+1) \cdot f(j+1, i) + f(j, i+1) = f(j+1, i+1)$$

and so the argument $\mathcal{I}(i+1, j+1)$ holds true.

Moving Forward

Lemma: We have proved that the product of n consecutive integers is divisible by $n!$.

Moving Forward

Lemma: We have proved that the product of n consecutive integers is divisible by $n!$.

Next steps (citing Manna & Pnueli):

- As mentioned in the beginning, process P_1 computes the numerator of the formula by successively multiplying into an integer variable, b, the factors n, $n-1$, ..., $n-k+1$. These factors are successively computed in variable y_1.

Moving Forward

Lemma: We have proved that the product of n consecutive integers is divisible by $n!$.

Next steps (citing Manna & Pnueli):

- As mentioned in the beginning, process P_1 computes the numerator of the formula by successively multiplying into an integer variable, b, the factors n, $n-1$, ..., $n-k+1$. These factors are successively computed in variable y_1.
- Process P_2, responsible for the denominator, successively divides b by the factors 1, 2, ..., k, using integer division. These factors are successively computed in variable y_2.

Algorithm Correctness

Citing Manna & Pnueli:

"For the algorithm to be correct, it is necessary that whenever integer division is applied it yields no remainder. We rely here on a general property of integers by which a product of m consecutive integers is evenly divisible by m!."

Algorithm Correctness

Citing Manna & Pnueli:

"For the algorithm to be correct, it is necessary that whenever integer division is applied it yields no remainder. We rely here on a general property of integers by which a product of m consecutive integers is evenly divisible by $m!$."

"Thus, b should be divided by y_2, which completes the stage of dividing b by $y_2!$, only when at least y_2 factors have already been multiplied into b by P_1. Since P_1 multiplies b by n, $n-1$, etc., and y_1 is greater than or equal to the value of the next factor to be multiplier, the number of factors that have been multiplied into b is at least $n - y_1$."

Algorithm Correctness

"Therefore, y_2 divides b as soon as $y_2 \leq n - y_1$, or equivalently, $y_1 + y_2 \leq n$. This condition, tested at statement m_1, ensures that b is divided by y_2 only when it is safe to do so."

Algorithm Correctness

"Therefore, y_2 divides b as soon as $y_2 \leq n - y_1$, or equivalently, $y_1 + y_2 \leq n$. This condition, tested at statement m_1, ensures that b is divided by y_2 only when it is safe to do so."

"The semaphore statements at l_1 and m_2 protect the regions $l_{2,3}$ and $m_{3,4}$ from interference. They guarantee that the value of b is not modified between its retrieval at l_2 and m_3 and its updating at l_3 and m_4."

Algorithm Formulation

Input: $k, n \in \mathbb{N}$, $0 \leq k \leq n$	
Output: $b \in \mathbb{N}$	
Locals: $y_1, y_2 \in \mathbb{N}$, mutex: r	
Initialization: $y_1 := n$, $y_2 := 1$, $b := 1$	
p1	p2
local t1: **integer**	**local** t2: **integer**
l_0 : **while** $y_1 > (n-k)$ **do**:	m_0 : **while** $y_2 \leq k$ **do**:
l_1 : **request** r	m_1 : **await** $y_1 + y_2 \leq n$
l_2 : $t_1 := b \cdot y_1$	m_2 : **request** r
l_3 : $b := t_1$	m_3 : $t_2 := b \; div \; y_2$
l_4 : **release** r	m_4 : $b := t_2$
l_5 : $y_1 := y_1 - 1$	m_5 : **release** r
l_6 :	m_6 : $y_2 := y_2 + 1$
	m_7 :

References

1. Nurdin Takenov (2017)
   "The product of n consecutive integers is divisible by n factorial"
2. Zohar Manna, Amir Pnueli (1995)
   "Temporal Verification of Reactive Systems: Safety"