A lower bound for adding \( n \) integers

**Problem**: adding \( n \) integers

- **Input**: \( x[0], \ldots, x[n-1] \) are integers (as large as \( n \) bits), initially in memory cells \( M[0], \ldots, M[n-1] \).
- **Output**: \( x[0] + \ldots + x[n-1] \) is left in \( M[0] \).

**Algorithm**: \( \text{parAdd} \)

**Theorem 14.13** Any Priority-Write PRAM with \( p \) processors that computes \( \text{parAdd} \) must do at least \( \lg(n) + 1 - \lg \lg(4p) \) steps.

**Corollary**: Any CREW PRAM, Common-Write PRAM, Arbitrary-Write PRAM, or Priority-Write PRAM that computes \( \text{parAdd} \) must do at least \((\log n)\) steps if \( p \) is bounded by any polynomial in \( n \).

### Sketch of proof

- \( n \) integers, each of \( n \) bits, are stored in \( M[0], \ldots, M[n-1] \)
- Simplification (to reduce the size of input and output space)
  - Input of \( \log(n+1) \) bits
  - Memory cells in \( M \) of \( \log(n+1) \) bits
  - \( p \) processors
  - \# of distinct inputs is \( 2^n \)
  - \# of possible different outputs is \( 2^p \)
  - \# of distinct states a processor can be in after \( t \) steps, with \( S_0 = 1 \)
  - \( v_t \): number of distinct values that could be in a memory cell after \( t \) steps, with \( v_0 = 2 \)
  - \# of processors
  - For \( t > 0 \) we have
    - \( v_t \leq 2v_{t-1} \)
  - For \( t > 1 \)
    - \( v_t \leq pS_{t-1} + v_{t-1} \)
    - \( v_t \leq (pS_{t-1} + v_{t-2}) \leq \cdots \)
    - \( v_t \leq (v_1)^{pS_t} \)
  - \( v_t \leq (v_1)^{pS_t} \leq (4p)^t \)
  - To distinguish \( 2^n \) possible different outputs, a parallel algorithm has to have \( v_t \geq 2^n \) after \( T \) steps.
  - \( 2^{\lg (v_t)} \geq 2^n \) after \( T \) steps.
  - Therefore, \( T \geq \frac{\lg(n) + 1 - \lg \lg(4p)}{\lg(v_1)} \) if \( p \) is bounded by a polynomial in \( n \).