444 We want to find the smallest number in $0,..n$ with property p. Linear search solves the problem. But evaluating p is expensive; let us say it takes time 1, and all else is free. The fastest solution is to evaluate p on all n numbers concurrently, and then find the smallest number that has the property. Write a program without concurrency for which the sequential to parallel transformation gives the desired computation.

§ We introduce array $A: [n*bin]$. We define the desired result R, condition $I\ i$, and helper specification P as follows.

$$R \;=\; \neg(\exists j: 0,..h'\cdot pj) \wedge (ph' \vee h'=n)$$
$$I\ i \;=\; \forall j: 0,..i\cdot Aj=pj$$
$$P \;=\; I\ n \wedge \neg(\exists j: 0,..h\cdot pj) \;\Rightarrow\; R$$

Now the program is

$$R \;\Leftarrow\; I0\Rightarrow I'n.\ \ h:=0.\ \ P$$
$$I0\Rightarrow I'n \;\Leftarrow\; \mathbf{for}\ i:=0;..n\ \mathbf{do}\ Ii\Rightarrow I'(i+1)\ \mathbf{od}$$
$$Ii\Rightarrow I'(i+1) \;\Leftarrow\; Ai:=pi$$
$$P \;\Leftarrow\; \mathbf{if}\ h=n\ \mathbf{then}\ ok\ \mathbf{else}\ \mathbf{if}\ Ah\ \mathbf{then}\ ok\ \mathbf{else}\ h:=h+1.\ \ P\ \mathbf{fi}\ \mathbf{fi}$$

The n iterations of the **for**-loop can be executed in parallel.

We can express the result of the sequential to parallel transformation at source as follows.

$$R \;\Leftarrow\; I0\Rightarrow I'n.\ \ h:=0.\ \ P$$
$$I0\Rightarrow I'n \;\Leftarrow\; i:=0.\ \ Ii\Rightarrow I'n$$
$$Ii\Rightarrow I'n \;\Leftarrow\; \mathbf{if}\ i=n\ \mathbf{then}\ ok\ \mathbf{else}\ Ai:=pi\ \|\ (i:=i+1.\ \ Ii\Rightarrow I'n)\ \mathbf{fi}$$
$$P \;\Leftarrow\; \mathbf{if}\ h=n\ \mathbf{then}\ ok\ \mathbf{else}\ \mathbf{if}\ Ah\ \mathbf{then}\ ok\ \mathbf{else}\ h:=h+1.\ \ P\ \mathbf{fi}\ \mathbf{fi}$$

To understand the execution, it might help to unroll the recursion a little: in the refinement of $Ii\Rightarrow I'n$, replace the recursive call $Ii\Rightarrow I'n$ by what's called **if** $i=n$ **then** ok **else** $Ai:=pi\ \|\ (i:=i+1.\ \ Ii\Rightarrow I'n)$ **fi**. And maybe do the same once more.