perplexity

i want to cover bunch of related problems connecting to each one that type of patterns

If you want to **master coding interviews by connecting related problems through shared patterns**, here's a step-by-step guide you can use for *any* major LeetCode pattern, especially for "Two Pointers," "Linked List," "Sliding Window," and more. This approach builds deep understanding, as you'll see how one problem leads directly to the next by tweaking constraints or objectives—turning isolated challenges into a network of related skills.

How to Connect Related Problems by Pattern

1. Identify the Core Pattern

For example, with "Two Pointers":

- Move two pointers towards each other
- Advance at different speeds
- Traverse two arrays/lists in parallel

2. List the "Starter" Problems

Start with a **basic example** under the pattern.

- Example: For Two Pointers, begin with "Two Sum (sorted array)."

3. Branch to Related Variations

After solving a starter problem, look for other problems with a similar skeleton, but different:

- Inputs (arrays, strings, lists, matrices)
- Constraints (sorted, rotated, duplicates, allowed modifications)
- Required output (indices, value, existence check, subarrays, etc.)

Each variant deepens mastery and exposes edge cases.

4. Create Chains of Connection

For every solved problem, ask:

- "What's the next logical twist?"
    - If you can solve "Two Sum," try "Three Sum."

    - Once you master removing duplicates from sorted arrays, try variants with at most K duplicates.
    - After cycle detection, try "Find starting node of cycle."
- Chart problems into a **learning tree**, where each node connects to a more complex or differently constrained problem.

5. Reinforce with Templates and Re-use

Build a flexible code template for the pattern. Adapt and reuse it, noting the tweaks for each new question.

Example Connection Map: Two Pointers

“Starter” Problem	Connected/Next Problem	New Twist
Two Sum (sorted)	Three Sum	Expand two pointers from pair to triplet
Reverse array	Reverse only vowels	Conditionally swap
Remove Duplicates (sorted array)	At most K duplicates	Extra counter/state
Is Palindrome (string)	Valid Palindrome II (one removal)	Allow one violation
Detect Cycle (linked list)	Find start of cycle	Track entrance, not just existence
Merge Two Sorted Lists/Arrays	Intersection of Two Arrays	Select only common elements
Move Zeros (array)	Sort Colors	Partition based on values
Longest substring w/o repeat chars	Min window substring	Introduce frequency tracking

Patterns You Can Chain in This Way

- **Two Pointers:** Connect palindrome, two-sum, three-sum, partition variants, and merging.
- **Linked List Patterns:** Start with reversal, cycle detection, go to reorder, merge, and complex variants (e.g., copy random pointers).
- **Sliding Window:** Begin with fixed-length window sum, move to variable size, max/min substrings, then constraints (unique elements, frequency).
- **Backtracking:** Simple permutations to combinations, then combinations with constraints.
- **Dynamic Programming:** Subset sum leads to coin change, then to partition, target sum, etc.

Recommended Method

- **Keep a table (like above) mapping each pattern to its variants and the logical step-ups.**
- After each problem, write “What next?” and seek out related LeetCode or similar problems.
- Use curated lists like LeetCode Explore, Sean Prashad’s Patterns, Grokking the Coding Interview to follow problem ladders.
- As you master a variation, move to the adjoining tougher or differently-configured question.
- Review and adapt templates; annotate where logic differs for each variant.

By practicing related problems in chains like this, you solidify your grip on core techniques and develop the foresight to instantly recognize and tackle new variations in coding interviews. If you want a table or resource list for a specific pattern, just let me know!