

IMPLEMENTATION OF ALGORITHMS – HEURISTICS: MEMORY HIERARCHY AND LOCALITY OF REFERENCE

LOCALITY OF REFERENCE(S)

- Locality of Reference(s)

- **The locus of data access** – and hence that of memory references – *is small at any point in execution* of a program:
 - Temporal Locality
 - Spatial Locality

LOCALITY OF REFERENCES

○ Locality of Reference(s)

- **The locus of data access** – and hence that of memory references – is small at any point in execution of a program:

○ Temporal Locality

- Data that are accessed (at a point in program execution) is likely to be accessed again in the near future:
 - i.e. same datum is likely to be repeatedly accessed in a short span during execution

○ Examples (of Manifestation of Temporal Locality)

- Instructions in the body of a loop
- Data (or a variable) that is computed iteratively
 - e.g. a cumulative sum or product

LOCALITY OF REFERENCES

○ Locality of Reference(s)

- **The locus of data access** – and hence that of memory references – *is small at any point in execution* of a program:

○ Spatial Locality

- Data that is accessed (at a point in program execution) is likely located adjacent to data that is to be accessed in the near future:
 - i.e. data accessed in a short span during execution is likely to be within a short region (in memory)

○ Examples (of Manifestation of Spatial Locality)

- Linear sequences of instructions
- Elements of Arrays (accessed sequentially)

LOCALITY INFLUENCES DESIGN

- A **Memory Hierarchy** amortizes cost in computer architecture:
 - fast (and therefore costly) but small-sized memory to
 - large-sized but slow (and therefore cheap) memory
- But a memory hierarchy is effective only due to **Locality** exhibited by programs (and the data they access)!
 - Longer the range of execution time of the program larger is the locus of data accesses:
 - this aligns with the memory hierarchy:
 - *increasing size with increasing access time of memory modules*

LESSON: IMPLEMENT PROGRAMS WITH LOCALITY!