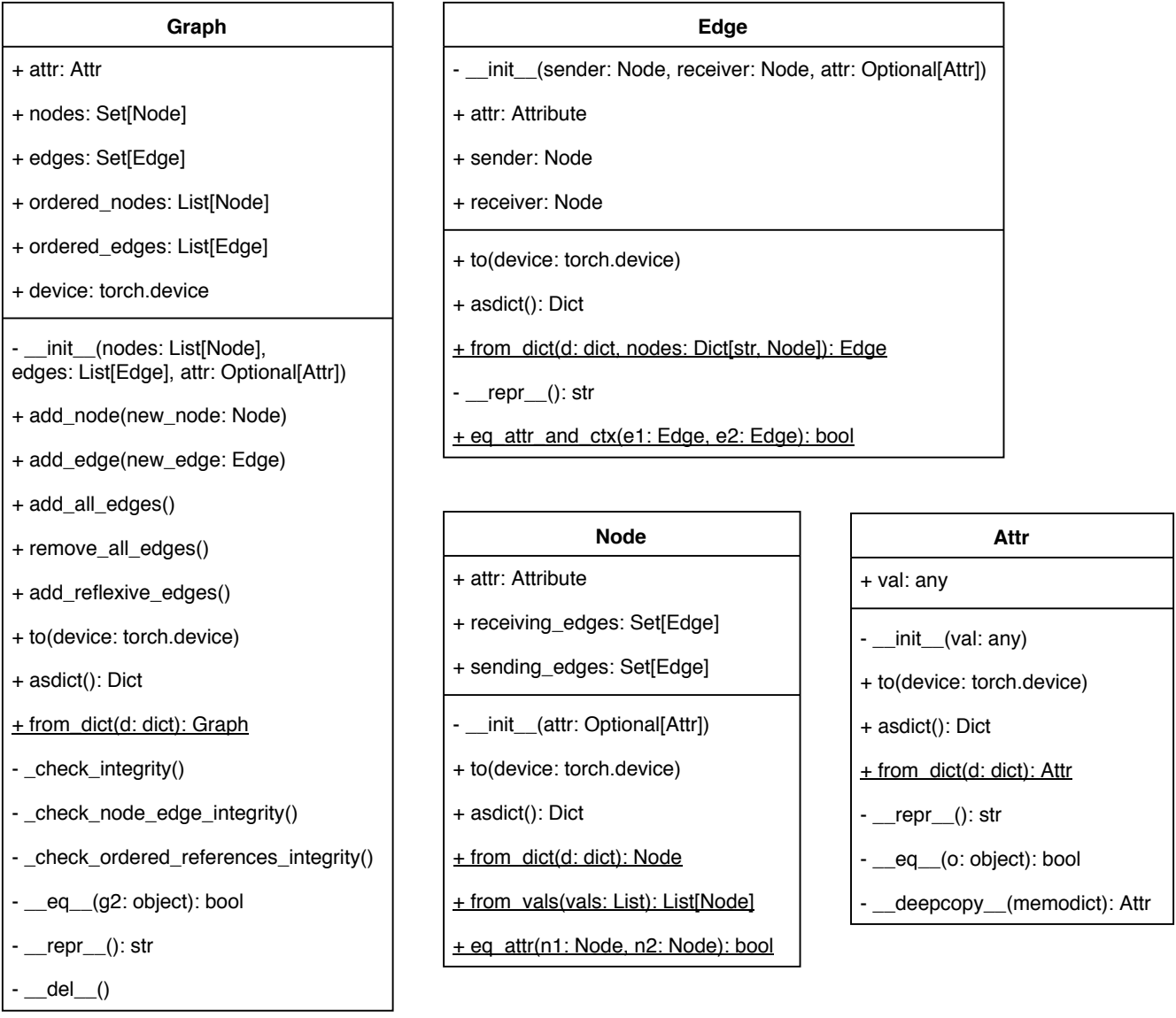
Graph
+ attr: Attr
+ nodes: Set[Node]
+ edges: Set[Edge]
+ ordered_nodes: List[Node]
+ ordered_edges: List[Edge]
+ device: torch.device
- __init__(nodes: List[Node], edges: List[Edge], attr: Optional[Attr])
+ add_node(new_node: Node)
+ add_edge(new_edge: Edge)
+ add_all_edges()
+ remove_all_edges()
+ add_reflexive_edges()
+ to(device: torch.device)
+ asdict(): Dict
+ from_dict(d: dict): Graph
- _check_integrity()
- _check_node_edge_integrity()
- _check_ordered_references_integrity()
- __eq__(g2: object): bool
- __repr__(): str
- __del__()
Edge
- __init__(sender: Node, receiver: Node, attr: Optional[Attr])
+ attr: Attribute
+ sender: Node
+ receiver: Node
+ to(device: torch.device)
+ asdict(): Dict
+ from_dict(d: dict, nodes: Dict[str, Node]): Edge
- __repr__(): str
+ eq_attr_and_ctx(e1: Edge, e2: Edge): bool
Node
+ attr: Attribute
+ receiving_edges: Set[Edge]
+ sending_edges: Set[Edge]
- __init__(attr: Optional[Attr])
+ to(device: torch.device)
+ asdict(): Dict
+ from_dict(d: dict): Node
+ from_vals(vals: List): List[Node]
+ eq_attr(n1: Node, n2: Node): bool
Attr
+ val: any
- __init__(val: any)
+ to(device: torch.device)
+ asdict(): Dict
+ from_dict(d: dict): Attr
- __repr__(): str
- __eq__(o: object): bool
- __deepcopy__(memodict): Attr