The Hungarian Algorithm or the Linear Assignment Problem? It finds maximum-weight matchings in bipartite graphs 📊. Wondering how it fits into Object Detection and Tracking?  
  
📍 𝗜𝗻 𝗦𝗶𝗺𝗽𝗹𝗲 𝗧𝗲𝗿𝗺𝘀:  
• This algorithm splits all bounding boxes (KF estimated boxes at frame T 🖼️ & detected boxes at frame T+1 🖼️) into 2 separate sets (bipartite graphs;).  
• "Maximum-weight" here refers to metrics like IOU, cosine similarity, or distance 📏.  
  
🎥 𝗩𝗶𝘀𝘂𝗮𝗹 𝗘𝘅𝗽𝗹𝗮𝗻𝗮𝘁𝗶𝗼𝗻:  
• 𝗙𝗿𝗮𝗺𝗲 𝟭: Detected boxes [A,B,C,D] & their KF estimates [A',B',C',D'] at time T.  
• 𝗙𝗿𝗮𝗺𝗲 𝟮: Detected boxes [A,B,C,E] at time T+1 & KF estimates [A',B',C',D'] from time T.  
🚫 𝗡𝗼𝘁𝗶𝗰𝗲: Detection of box D is lost in Frame 2. Reasons? Maybe the object isn't there, or detection failed due to occlusion or low confidence. Plus, a new object E is detected!  
• A Bipartite graph helps us see this. The edges? They represent metrics like IOU.  
💡 For instance, cost from A' to A is the highest among all pairs ([A', A],[A',C],[A',E],[A',B]). So, A' and A are the same object instances across frames!  
  
𝗥𝗲𝗺𝗲𝗺𝗯𝗲𝗿:  
• KF = Kalman Filter 🌀 an algorithm for precise state estimation.  
• A Bipartite Graph 📊 divides its vertices into two separate groups. The catch? No two vertices in the same group can connect directly to each other!  
• Hungarian Algorithm matches boxes. But, overall tracking algorithms consider KF estimates from previous frames & detected objects from the current one. This way, they know if an unmatched box is a new object at frame T+1 or a missed object from frame T

