4 货币顺序的确定

在这一节中，我们在第4.1节中概述了所提出的货币价值计算方法，并在第4.2节中讨论了关于货币顺序结果的可靠性评估。

4.1 货币价值计算算法 由于CC描述了几个目标属性的值之间的部分顺序，所以一些图元之间的货币顺序仍然无法推断。没有从CCs中推理出任何货币顺序的数据是很难被评估为货币的。这促使我们确定Gc中所有节点的货币顺序，并实现所有图元的总货币顺序。有了给定的CC，我们可以将货币顺序图Gc所表达的原始部分顺序细化为一个衬垫顺序。在某种程度上，当真正的时间戳不可用时，货币顺序是一种时间戳的替代。因此，数据的时间顺序被揭开了，度量标准帮助数据质量解决了一致性和完整性的问题。我们解决方案的一般原则是以与图结构兼容的方式将区间（0,1）中的不同数字均衡地分配给货币顺序图Gc中的所有节点。这些分配的数字被认为是定义在图元上的货币价值，用Curr(v)表示。一个直观的货币价值分配方法是对货币订单图Gc进行拓扑排序，并获得这样的价值。然而，拓扑排序结果[13]对于计算Gc上所有顶点的货币顺序来说，并不总是足够稳定和准确。在此，我们提出一种货币顺序确定方法，以实现统一和准确的货币顺序确定。

在货币顺序图中，同一实体的可比较货币的图元构成了路径。这有助于确定图中的货币价值。我们认为图Gc中的任何一条路径都是一个货币订单链，用S¼ fv1; . . ; vmg表示，其中8 vk 2 S; k 2 ½1; mÞ, vk ? vkþ1。据此，对该方法的概述表述如下：①我们首先添加一个全局性的起始节点s，该节点与所有没有入边的图形节点相连，并添加一个全局性的终端节点t与所有没有出边的节点相连。然后s得到时间戳0，t得到时间戳1；ii）重复地选择一个没有时间戳的最大货币秩序链。然后，新的时间戳被分配给这些节点，其方式是链中相邻节点之间的时间差是相等的。

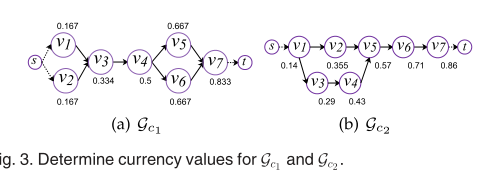
在该方法的解决方案中，有一件关键的事情需要考虑。为了使等距分配可行，链中的所有节点必须从其前任和后任继承了对可能的时间戳范围的相同约束。因此，我们需要找出两个节点之间在Gc上具有最长深度的最大货币订单链Smax，然后将货币值CurrðvÞ分配给Smax中的每个节点。为了计算节点v的前任和继任者的可能价值范围，我们在定义7中引入了v的货币订单边界，分别用infðvÞ和supðvÞ描述v的潜在最小和最大价值。界限以及节点在货币订单链中的位置对于保证每个节点的价值范围的准确性非常重要。因此，我们在图上定义有效边，以确定是否应该选择一条边来形成最大的货币订单链。这个阶段将在第4.1.1节讨论。

定义7（货币次序界线）。在确定货币价值时，S ¼ fv1;. . ; vmg中节点vi的上界和下界被定义为。(a) vi的币序上限是supðviÞ ¼ min Curr(vi?)，其中vi?代表与vi相连的任何一个子节点；(b) vi的币序下限是infðviÞ ¼ max Curr(v?i)，其中v?i代表与vi相连的任何一个祖先节点。有了边界值，我们需要发现两个节点之间的最大货币顺序链。值得注意的是，这种发现方法与简单地找出图上最长的路径不同。由于新的货币价值是根据之前确定的价值分配的，所以我们需要发现两个节点之间的最大货币顺序链，这些节点的货币价值现在是已知的。此外，我们还需要保证Smax具有正确的货币订单边界。我们提出了关于有效货币订单链的定义，然后介绍了发现这种链的算法。这一部分将在第4.1.2节讨论。

算法1. CurrValue 输入：实体E的货币图Gc ¼ ðV; EÞ 输出。1 在Gc中加入s和t，让s指向所有0内度的边，t指向所有0外度的边；2 初始化CurrðsÞ ¼ supðsÞ ¼ infðsÞ 0，CurrðtÞ ¼ supðtÞ ¼ infðtÞ 1。3 while 9 Curr(vi)尚未确定 do 4 复制Gc并得到其转置图GT c; 5 Updatebound(GT c , sup, < ); 6 Updatebound(Gc, inf, > ); 7 for eði; jÞ 2 E do 8 根据定义8对eði; jÞ的验证进行标注; 9 恢复涉及的节点; 10 S getMaxS(Gc), k jSj; 11 Value inf(S½1? ), Inc supðS½k?Þ?infðS½1?Þ k?1 ;12 foreach h从2到k ? 1 do 13 Curr(vh) Value + Inc; 14 Value Curr(vh); 15 return Gc;

算法1显示了整个货币价值计算过程。我们首先将上述的全局开始和终端节点s和t到Gc，以确保货币价值位于（0,1）。我们更新图上节点的货币顺序边界，并发现有效的边（第5-9行），其中我们记录了Gc的副本，并首先用Gc的转置图更新Gc中节点的sup，然后在Gc上更新inf。(详见4.1.1节）。之后，我们在第10行找到目前最长的候选链S（详见算法3），其中k ¼ j S j是S中元素的数量。我们为S中的每个v分配规范化的货币值（第8-11行）。随着界限的确定，我们使用S中inf(S½1?)(resp. sup(S½k?))的下限(resp. upper)来计算S中所有元素的货币价值。 我们迭代地运行上述步骤，直到所有节点上的货币订单被计算出来。

1. 我们现在计算Gc1和Gc2中的货币价值。我们首先找出图3a中的Smax ¼ fs; v1; v3; v4; v5; v7; tg。对于Smax中的每个节点，Curr(viþ1) = Curr(vi) + supðvkÞ?infðv1Þ k?1，k ¼ 7（包括s和t）。然后，我们可以很容易地确定Curr(v2)=0.167，Curr(v6)=0.667。对于图3b中的Gc2，我们首先找到Smax ¼ fs; v1; v3; v4; v5; v6; v7; tg，然后计算Curr(vi) t o b e f0; 0:14; 0:29; 0:43; 0:57; 0:71; 0:86; 1g。此后，只有v2的货币价值没有确定。我们用supðv5Þ和infðv1Þ得到Curr (v2) = 0.355。



4.1.1 更新边界和有效边缘 图上的每条链都揭示了根据CCs推导出的传递性货币关系的长度。在每次迭代过程中，并非所有边都有助于Gc中的Smax发现。我们应该在币值计算过程中，通过计算v的货币顺序界线来确定节点v是否可以补足Smax。为了在算法1的每次迭代过程中区分出现在的最大货币订单链Smax，我们需要确定能构成Smax的边。我们认为这样的边是有效的边。我们在定义8中定义了边的有效性。候选的Smax存在于这种由有效边形成的货币订单链中。因此，可以根据这些有效边有效地找到Smax

定义8（边的有效性）。在三种情况下，一条边eði; jÞ 2 EðGcÞ是一条有效的边。情况1. 如果Curr(vi)和Curr(vj)都没有确定，那么eði; jÞ是一条有效边，即supðviÞ = supðvjÞ，infðviÞ = infðvjÞ。情况2. 如果Curr(vi)确定，Curr(vj)不确定，那么eði; jÞ是一条有效边，即infðviÞ = infðvjÞ。情况3. 如果Curr(vj)确定，Curr(vi)不确定，eði; jÞ是一条有效边，即supðviÞ = supðvjÞ。请注意，如果Curr(vi)和Curr(vj)都被确定，eði; jÞ不是有效边，因为vi和vj在之前的迭代中已经被访问过。由于他们的货币值已经得到，eði; jÞ在本步骤中是无效的。在更新有效边时，我们使用vi及其确定的Curr(vi)来更新vi所能到达的节点的inf(vi?)和sup(vi? )。这两个边界都是通过函数Updatebound更新的。我们以算法2中的sup为例。我们递归地选择内度为0的vi，列举所有vi? 并比较vi和vi? 的sup值。如果supðvi? Þ>supðviÞ，那么sup(vi?)就会被更新为sup(vi)的值。在所有eði;? Þ被处理后，我们暂时从V中删除vi。再来看看算法1的第4-6行。我们执行函数 Updatebound(G, bound, op)两次，分别更新sup和inf。在第9行中，我们在有效边被标记后恢复Gc上的所有节点，并继续以下步骤。

1. 我们讨论另一个货币图Gc3，以清楚地表明我们的方法如何在具有复杂货币关系的图元上工作。图4a显示了Gc3的第一条最长的链Smax1。有了确定的Curr(vi) ði 2 ½1; 7?Þ，我们就可以对其余节点v8; . . . ; v12的边界进行分析，并找到下一个Smax。由于v8; v9; v10; v11都能到达v6，而v6在它们的子孙节点中具有最小的货币价值，所以supðv8; v9; v10; v11Þ ¼ Currðv6Þ ¼ 0:75. v12只到达v7，所以supðv12Þ ¼ Currðv7Þ ¼ 0:875. 同样地，图4b中v1可以从v8和v9到达，而v10；v11；v12可以到达v2。因此，infðv8; v9Þ ¼ Currðv1Þ ¼ 0:125, infðv10; v11; v12Þ ¼ Currðv2Þ ¼ 0:25。v8; v9; v10; v11的货币值不确定，v8; v9 (resp. v10; v11)有相同的界限。根据定义8中的情况1，eð1; 8Þ; eð2; 10Þ, eð8; 9Þ和eð10; 11Þ有效。所有有效的边在图4c中用橙色标记

4.1.2 发现最大链 在更新了边界和有效边之后（在每次迭代中），我们开始找出最大的货币订单链。为了保证链中相邻节点之间的输出货币价值差异相等，我们需要在由有效边连接的节点中发现Smax。我们认为这种链是定义9中定义的有效货币订单链。然后，我们的任务是找出一个具有最长深度的有效货币订单链

算法2. Updatebound(Gc, bound, op) 输入。Gc, bound 2 fsup; infg, op 2 f < ; > g 输出：更新的Gc与bound 1 while V 6¼ ; do 2 foreach eði; jÞ 2 E with 0 in-egree do 3 if bound½vi? op bound½vj? then 4 bound½vj? bound½vi?; 5 V V nvi;